Privacy-preserving Metrics for an mHealth App in the Context of Neuropsychological Studies

Alexander Gabel[1][a], Funda Ertas[2][b], Michael Pleger[1][c], Ina Schiering[1][d] and Sandra Verena Müller[2][e]

[1]*Faculty of Computer Science, Ostfalia University of Applied Sciences, Salzdahlumerstraße 46/48, Wolfenbüttel, Germany*

[2]*Faculty of Social Work, Ostfalia University of Applied Sciences, Salzdahlumerstraße 46/48, Wolfenbüttel, Germany*

Keywords: mHealth, Data Minimization, Privacy by Design, Privacy by Default, Data Aggregation, Metrics, Neuropsychology, Empirical Study.

Abstract: The potential of smart devices as smartphones, smart watches and wearables in healthcare and rehabilitation, so-called mHealth applications, is considerable. It is especially interesting, that these devices accompany patients during their normal life. Hence they are able to track activities and support users in activities of daily life. But beside the benefits for patients, mHealth applications also constitute a considerable privacy and security risk. The central question investigated here is how data about the usage of mobile applications in empirical studies with mHealth technologies can be collected in a privacy-friendly way based on the ideas of Privacy by Design. The context for the proposed approach are neuropsychological studies where a mobile application for Goal Management Training, a therapy for executive dysfunctions, is investigated. There a privacy-friendly concept for collecting data about the usage of the app based on metrics which are derived from research questions is proposed. The main ideas underlying the proposed concept are a decentralized architecture, where only aggregated data is gathered for the study, and a consequent data minimization approach.

1 INTRODUCTION

The use of mobile applications in healthcare and rehabilitation has in general a huge potential. Especially smart devices as smartphones, smart watches, smart glasses and wearables are widely used in health-related research projects in areas as diagnosis, therapy and rehabilitation (Garcia-Ceja et al., 2016; Grünerbl et al., 2015; Jamieson et al., 2019). Since these smart devices accompany patients in daily life, data about the behavior of patients can be tracked and patients could be supported in their daily activities or they can be coached to lead a more healthy life.

But on the other hand in health apps substantial security and privacy issues were detected (Papageorgiou et al., 2018; Huckvale et al., 2015) and in studies about willingness of users to share data (Di Matteo et al., 2018; Beierle et al., 2019) users stated privacy concerns, especially since data transfer and usage is not sufficiently transparent.

Concerning privacy and data protection in the European Union compliance with the General Data Protection Regulation (GDPR) (European Union, 2016) is required. The basis of this regulation is summarized in the principles of data protection in Article 5 of GDPR as lawfulness, fairness, transparency, purpose limitation, data minimisation, accuracy, storage limitation, integrity, confidentiality and accountability.

A central approach to integrate privacy early in the development process of new systems resp. services are *Privacy by Design* and *Privacy by Default*. To model privacy risks, *privacy protection goals* (Hansen et al., 2015) confidentiality, integrity and availability, transparency, unlinkability and intervenability are helpful. To integrate privacy requirements in system architectures *privacy design strategies* (Colesky et al., 2016) summarizing the important concepts minimize, hide, separate, abstract, inform, control, enforce, demonstrate can be used.

These Privacy by Design methodologies were used in the context of the research project SecuRIn

[a] https://orcid.org/0000-0002-2660-4350
[b] https://orcid.org/0000-0002-7940-1178
[c] https://orcid.org/0000-0001-5398-3694
[d] https://orcid.org/0000-0002-7864-5437
[e] https://orcid.org/0000-0003-4450-9936

Gabel, A., Ertas, F., Pleger, M., Schiering, I. and Müller, S.
Privacy-preserving Metrics for an mHealth App in the Context of Neuropsychological Studies.
DOI: 10.5220/0008982801660177
In *Proceedings of the 13th International Joint Conference on Biomedical Engineering Systems and Technologies (BIOSTEC 2020) - Volume 5: HEALTHINF*, pages 166-177
ISBN: 978-989-758-398-8; ISSN: 2184-4305

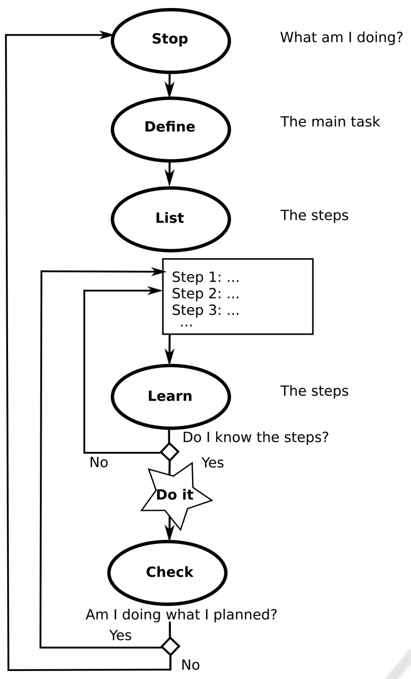


Figure 1: Goal Management Training.

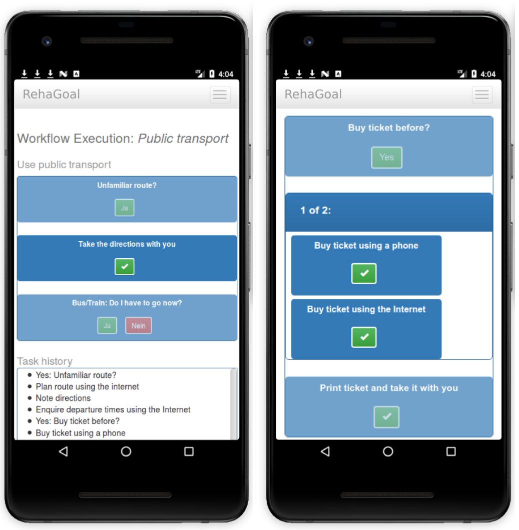

RehaGoal
Workflow Execution: Public transport
Use public transport
Unfamiliar route?
Ja
Take the directions with you
Bus/Train: Do I have to go now?
Ja
Nein
Task history
Yes: Unfamiliar route?
Plan route using the internet
Note directions
Enquire departure times using the Internet
Yes: Buy ticket before?
Buy ticket using a phone
RehaGoal
Buy ticket before?
Yes
1 of 2:
Buy ticket using a phone
Buy ticket using the Internet
Print ticket and take it with you

Figure 2: Screenshots of Individual Workflow.

where a mobile application for the so-called Goal Management Training (GMT) (Figure 1) (Levine et al., 2000; Bertens et al., 2013; Stamenova and Levine, 2018), an important therapy for executive dysfunctions, was developed (Gabel et al., 2018; Müller et al., 2019).

Executive dysfunctions are deficits of brain-damaged patients concerning "the selection and execution of cognitive plans, their updating and monitoring, the inhibition of irrelevant responses and problems with goal-directed behaviour usually result in disorganized behaviour, impulsivity and problems in goal management and self-regulation" (Emmanouel, 2017, p. 17).

In the mobile application RehaGoal which is realized as a web app, therapists can model individual workflows in a visual editor (Eckhardt et al., 2019) to help patients in activities of daily life as e.g. taking public transport, shopping or cleaning activities. Workflows can then be executed on a smartphone (optionally coupled with a smart watch), which will always display the current task, and can remind the user regularly (Figure 2). Each task can contain a textual description and an image. In addition there are elements as e.g. loops, yes/no questions, choices. To further assist users, a text-to-speech (TTS) engine can be enabled to read texts aloud. Workflows can be arranged in a so-called schedule, such that multiple workflows may be executed in a sequence.

The application is used in a neuropsychological intervention study and is compared with classical goal management training. As pre/post tests are not able to provide data during most time of the study, and self-reports by patients can be biased, it is intended to additionally collect data on the device in an automated manner. Since the behavioral data involved may be sensitive and may allow tracking, it is important that this is done in a privacy-preserving way, following Privacy by Design and data minimization principles.

With the introduction of the RehaGoal mHealth app to the planned intervention study, new possibilities regarding data collection were discussed, which in the long-term could assist in therapeutic use-cases. In addition to the pre/post tests as a common practice in intervention studies, the use of mobile applications allows to collect data from the day-to-day behavior of the patients. On one side this allows additional insights about the success of the therapy, as well as the app usage, but on the other side this also might entail significant privacy risks, as the device is potentially able to track a large part of the users lifes. To address these risks, the development process of mobile applications was integrated in a Privacy by Design process (Gabel et al., 2018). One aspect of this Privacy by Design process, the concept for privacy-preserving metrics to measure the impact of the mobile application, is considered here.

In general the correlation between app usage and test performance, as well as the usability of the app was of interest. One of the goals was to minimize the data collection needed for answering those questions, without losing too much data quality. Therefore one privacy-related question was, whether it is possible to formulate research questions about the data, before actually having collected the data in the context of a study. In many cases, foremost in big data analysis, the questions are asked after (raw) data is already available, or at the very least raw data is still available in order to reformulate questions or ask additional ones. However, in order to take data minimization a step further and reduce the possibility of additional inferencing based on the collected data, following purpose limitation, data was aggregated on the device where it is collected, i.e. before data is transferred to the study provider. This meant, that much care had to be taken in formulating the research questions and precisely checking, which data is needed in order to get insight.

In this paper a Privacy by Design approach was applied to develop metrics for an empirical study based on the mobile application described above. To this aim research questions were formulated before the study. Based on the principle of data minimization a metric language was derived from the research questions. For the integration of metrics in the mobile application, a metric architecture is proposed. The approach presented here has a focus on data minimization via aggregation and is based on the concept of decentralization to foster privacy (Troncoso et al., 2017).

The rest of the paper is organized as follows: In section 2 an overview of existing approaches and issues is given, section 3 summarizes the methodology. In the following the concept for privacy-preserving metrics is described. First research questions and derived metrics are stated in section 4, afterwards the metric language and the corresponding metric architecture is presented (section 5). Afterwards preliminary results of the approach are summarized (section 6).

2 RELATED WORK

Data collection using smartphones is already widely used in mHealth. Use-cases can be classified by whether they are using active data, requiring active participation (e.g. surveys), or passively collected data (i.e. sensor data) (Onnela and Rauch, 2016). Furthermore there are applications which focus on classical medicine, such as measuring the blood glucose level (Cafazzo et al., 2012), as well as ones which are located in the psychological area of research. The latter applications often use data collected by smartphones to measure behavior, e.g. regarding to Digital Phenotyping (Onnela and Rauch, 2016; Kleiman et al., 2018), or to gather insight on aspects of the mental state of a person (Grünerbl et al., 2015; Garcia-Ceja et al., 2016). These approaches typically collect a large amount of passive data and use active data, as well as psychological tests as ground-truth data. Using methods from the area of data mining and machine learning, classifiers are trained, to inference mental state from certain behaviors. This has been used to e.g. detect the manic/depressive state of people with bipolar disorder (Grünerbl et al., 2015), as well as stress detection (Garcia-Ceja et al., 2016). Often the data is first transformed by feature engineering & extraction using domain specific knowledge. While there is research towards privacy-sensitive deep learning techniques, e.g. based on differential privacy (McMahan et al., 2018) or federated learning approaches using a cryptographic protocol for secure aggregation (Bonawitz et al., 2017), deep learning is still not used as often, due to e.g. overfitting at smaller sample sizes (Mohr et al., 2017). However, publications often do not state, whether features are designed before the actual data collection or afterwards, as well as when, where (which device) and how often the features are calculated on the raw data. All these factors potentially have privacy implications. The amount of information which may be inferenced from raw data is much harder to approximate than for specially designed features. Privacy by Design and Data Minimization are therefore not always implemented as intended by the GDPR (European Union, 2016). For example (Grünerbl et al., 2015) designed a smartphone logging application, which collects information about phone calls, sound & voice features to e.g. detect emotional state, physical motion (accelerometer), as well as travel patterns (GPS). The patient may decide for every day whether she/he is comfortable with storing the data. No voice recognition was performed, phone numbers where anonymized and GPS tracks were transformed to an artificial coordinate system, where $(0,0)$ always indicates "home". Some requirements (e.g. regarding the GPS tracks), were later demanded by an ethics board instead of integrating Privacy by Design in the design process of the application. It is not clear, whether the raw data is stored on the phone or when it is aggregated. In the worst case, the smartphone may collect raw streams of sensor data and later on researchers "anonymize" this raw dataset, which includes computing the features. In contrast a much better way would be to calculate the features locally on the smartphone, while storing as little temporary raw data as possible. Many features may also be calculated directly from the raw data stream, i.e. only storing intermediate values for aggregation.

Another area, where sensor data is collected on mobile devices, are so-called fitness apps. For these applications often existing frameworks are used as e.g. Google Fit SDK[1], Apple HealthKit [2], Samsung Health SDKs [3], or Garmin Health API [4]. These typically include APIs for recording data from sensors, including data types for a common representation of units, as well as interfaces for querying data sets. The basic architecture mainly relies on storing raw data on provider-specific cloud services. Aggregates may be computed afterwards based on this raw data. Most services furthermore require the data to be stored on their cloud service (Google, Samsung: data is synchronized with server, Garmin: Data needs to be uploaded to Garmin Connect), with Apple HealthKit being the only exception (data is kept locally on the user's device). Furthermore in all cases accurate timestamps (often millisecond precision) are always included for every event. Samsung also includes a unique device identifier (UUID) in every event. Regarding data minimization, it would be preferable to only collect and store aggregate values, to specify the needed accuracy (i.e. bucketing), and only include additional data if required for the specific use case. For example timestamps are most likely not required to milli- or nanosecond (Google) precision. Furthermore e.g. step counts aggregated into minute buckets or smaller time steps might also be unnecessary for most use-cases.

There have also been several analyses regarding the privacy and security of mHealth apps from app stores (Martínez-Pérez et al., 2015; Knorr and Aspinall, 2015; Morera et al., 2016; Treacy and McCaffery, 2016; Papageorgiou et al., 2018). Huckvale et al. (Huckvale et al., 2015) assessed several aspects of accredited health and wellness apps, including data entered into the app, whether data is transmitted to online services and whether it is encrypted, but also other properties, such as the privacy policy or operating system permissions. Two-thirds of the apps collected strong identifiers (which enable linkability) and 71 % had a mandatory registration process, while half captured weak identifiers and 57% recorded potentially sensitive information. It is at least questionable, whether a Privacy by Design process was applied in the development, as strong identifiers (email address, full name, etc.), as well as a registration could possibly be avoided in a minimal use-case of most apps, i.e. without sharing data with other parties following the principle of data minimization. Concerning security vulnerabilities, Huckvale et al. furthermore discovered insecure data storage, data leakage or weak server-side controls, as well as insufficient connection encryption. He et al. found similar issues (He et al., 2014) and noted, that in the case of using third party servers, such as Amazon Web Services, it is mostly unclear, whether data is stored in an encrypted fashion, such that third parties do not have access to potentially sensitive data. End-to-end encryption is not mentioned, which however could strongly improve the users' control over their own data, including the decision, whether to actually share data and with whom. Goyal et al. specifically assess wearable health trackers (Goyal et al., 2016). Recently Papageorgiou et al. also found various major issues in mHealth applications, including transmission of identifiers, personal or health data to third parties, sometimes even unencrypted. Mense et al. discovered similar issues, particularly with third-party advertising or analytics solutions (Mense et al., 2016). This sometimes includes GPS coordinates, age (group) & gender, as well as health-related data, such as heart rate. Additionally strong identifiers including email address and device IDs are transmitted to the application developer's website.

3 METHODOLOGY

In the interdisciplinary team consisting of neuropsychologists and computer scientists research questions were identified and discussed based on experiences from pilot studies and questions about the use of features of the mHealth application. This was an iterative process, in which abstract topics were broken down step by step into separate questions, for which data on the devices could be collected. For a better overview research questions are grouped according to research motivation (Table 1).

Ideas to measure the part of the patients behaviour which is important for the study were validated based on the question whether the amount of data is necessary in proportion to the intended aim. The corresponding terminology of the approach is summarized in Table 2.

The amount of data and needed granularity was expressed in the form of *metrics* (Table 3). A scheme was developed for formulating metrics. In this context especially the concept of an adequate granularity was important. The mobile application allows to schedule workflows consisting of tasks. Hence it is possible to measure metrics based on tasks, single executions of workflows, aggregate over all executions of a workflow or over schedules which comprise several workflows. It is possible to aggregate data at more than one of these levels. Therefore this workflow based granularity was named *assignment*.

[1]https://developers.google.com/fit/

[2]https://developer.apple.com/documentation/healthkit

[3]https://developer.samsung.com/health

[4]https://developer.garmin.com/health-api/overview

To describe the exact moment when the measurement should take place during the execution of the software, e.g. at the start of the execution of a workflow, a so-called *record point* needs to be stated, that represents this point of time in the software.

Furthermore it needs to be specified, if several measurements should be aggregated or if single measurements are captured and stored. Concerning the aggregation, also the aggregation function needs to be stated. In the following this methodology is applied to the research questions of studies in the context of the mHealth application considered here. Each research question could be described by one or more metrics formulated based on this scheme (section 4). Metrics and the metric scheme were used as the basis of a metric language and a flexible architecture based on the metric language is described (section 5).

4 RESEARCH QUESTIONS & METRICS

As a first step research questions of the interdisciplinary team were developed (Table 1). Motivations for research questions are to detect correlations between the results of the neuropsychological tests, answers in questionnaires, interviews and the use of the mobile application (*therapeutic results*), usability of the mobile application and usage of features in workflow design (*usability*), implications of the specific design of workflows (*workflow design*). Since some research questions are related to more than one motivation, they are categorized as follows:

- Therapeutic results
- Usability & Therapeutic results
- Usability & Workflow design
- Usability

Because in the context of the study the usage of workflows by participants and therefore the results of metrics highly depend on the modeling of particular workflows, also workflows, potentially in different versions are stored, including pictures, such that in the evaluation of the study problems such as potential modeling and design issues can be understood. Every modification of a workflow therefore gets assigned its own workflow version identifier, which can be used to group metrics for a specific version.

Some questions have multiple metrics assigned, as there may be different time granularities, aggregate functions, events (when to record the metric), or obligatory private metrics recording temporary data necessary for answering them. Table 1 gives an

Table 1: Research questions and associated metrics.

Category	Research question	Metrics
Therapeutic Results	Does the number of workflow executions correlate with changes in the goal attainment scale / neuropsychological tests / subjective well-being?	**m1**
	Does the repeated application of a workflow correlate with less assisted executions over time?	m2, m2a
	Does the number of completed workflow executions correlate with changes in the goal attainment scale / neuropsychological tests / subjective well-being?	**m3**
	Does the repeated completion of a workflow correlate with less assisted executions over time?	m4, m4a
	Does the number of canceled workflow executions correlate with changes in the goal attainment scale / neuropsychological tests / subjective well-being?	m5
	Does the repeated cancellation of workflows correlate with less assisted workflow executions over time?	m6, m6a
	Does the repeated completion of a workflow correlate with the time taken for the execution of it?	m7
	How does the amount of reminders per task change over consecutive executions?	m9, m9a, m9b, m9c_private
Usability & Therapeutic Results	Does the usage of TTS correlate with the number of completed/-canceled workflows?	**m17** – m21
Usability & Workflow Design	What could be possible reasons for the active cancellation of a workflow?	**m8**
	How far are workflows executed when they are canceled? Are workflows canceled at specific tasks?	m22
	Does the type of presentation correlate with changes in the time taken for completing a task?	m23 – m28
Usability	Are reminders closed once the task has been completed or as soon as the dialog appears on the device?	**m10, m11, m11_private**
	How often was the scheduling feature used (and therefore possibly better integrated)?	m12
	Is the scheduling feature canceled less over time? (and therefore possibly better integrated)?	**m13**
	How are schedules used? How many (different) workflows are scheduled?	m14
	How are schedules used? How many workflows were executed before canceling a schedule?	m15
	How is the distribution of the number of workflows in completed schedules?	m16
	How many (different) workflows are executed in a completed schedule?	m15, m16

Table 2: Terminology.

Term	Explanation
Assignment	*Inside a metric definition*: The names of the keys under which values of a metric should be grouped. For example `['workflow', 'execution']` would mean that the metric records separate snapshots for every combination of workflow(Id) and execution(Id). Furthermore these concrete values are stored in the corresponding snapshot. *In the context of a record point*: A mapping of (at least) all assignment keys in metric definitions for this record point to the value for the current event. For example {`workflowId: '123-456', executionId: 42`} would be a valid assignment for the assignment definition above.
Execution	A workflow is being executed, when it is being performed by a human assisted by the application. From the start of a workflow to the completion or cancellation counts as being part of the execution.
Metric	Measures or computes a certain value in the context of a given assignment. Metrics may aggregate several measurements into a single value, store each measurement separately and can also trim the accuracy of a value before it is stored.
Metric Type	Currently four different types are defined: number metrics (**int**eger and **float**ing point metrics) measuring primitive values, **dur**ation metrics measuring the time difference between two record points, and **meta** metrics, which compute a value based on another metric
Recording	A metric is recorded when it is triggered by a record point and measures or computes a value which is then stored in the metric database.
Record Point	Named event in the program source code which may trigger the recording of several metrics. Apart from a required assignment, an optional (dynamic) value may be provided.
Schedule	A schedule consists of multiple workflows which should be executed in a given order. It can be created dynamically by the user when needed.
Snapshot	A snapshot consists of one or more measurements aggregated into a single value, sometimes including additional information necessary for updating the aggregate or (trimmed) timestamps. Furthermore each snapshot stores its assignment and a sequential index.
Task	Simple step in a workflow, which is not being broken down into steps any more.
Workflow	Representation of a task consisting of many steps performed by human, modeled using a block-based visual programming language

overview about research questions and the derived metrics used in our intervention study.

A selection of these metrics (marked as bold) used to illustrate the concept is detailed in Table 3. The metric definition scheme developed here consists of the following parts:

- #: Metric ID
- *Subject:* Description of the metric
- *Type:* Determines what type of value should be recorded.
- *Assignment:* describes for which identifiers data may be collected and potentially aggregated. The columns correspond to *schedule* (s), *workflow* (w), *workflow version* (v), *execution index* (e) and *task index* (t). These are arranged in a kind of hierarchy, i.e. a schedule can contain multiple workflows, a workflow may have multiple versions, a version may be executed more than once and during execution different tasks occur. Metrics can have an assignment which covers multiple areas, e.g. it could be of interest to collect information about sequential executions of a particular workflow (w,e), but it may also be useful to instead find out about sequential executions per user. A `global` assignment in the table actually corresponds to an assignment per device/user, i.e. no assignment in the other categories.
- *Record Point:* named events in the program source code which should trigger the recording of the metric
- *Aggregate:* Aggregation function (*op.*) and aggregation interval (*time*) (if applicable)
- *#Sn:* number of metric results that are collected, denoted as snapshot count
- *Options:* More specific options, e.g. accuracy

Based on the emphasized metrics this metric scheme is explained in more detail. The goal of metric *m1* is to count the number of workflow executions for each workflow separately. In the example workflows are denoted by a title explaining the aim of the workflow instead of the workflow ID. As a potential result the workflow “Make coffee” may have been executed only once, while “Write a letter” may have been executed 12 times. Since the result of the metric is a natural number, the type of the metric would be `int`. The metric should be increased by one when the record point `workflowStart` is reached in the mobile app.

To count instead only workflow executions that are completed (*m3*), instead the record point `workflowEnd` is used. To count values, the aggregate operation `sum` adds the constant 1 to the metric value, every time the event described by the record point occurs (*Option* `constValue: 1`). Since it is not intended to measure a progression over time in this metric, the value is aggregated over the whole study (measurement period) which is denoted by *time* `all` time.

Another example is metric *m8*, which measures the duration of workflow execution executed before it was actively canceled by the participant. In this case the type is duration `dur`, and the time interval from `workflowStart` to `workflowAbort` is measured. Since cancellation of workflows is very important in the context of the corresponding reserach question, a separate value for each workflow execution is collected, i.e. no aggregation is applied. In this case the assignment in the metric scheme is `workflow` and `execution`. Since there is no limit for the number of values, the snapshot count is infinite.

A more complex construct of a so called *meta metric* is a metric whose values are calculated based on the values of a second metric as e.g. *m11* based on *m11_private*. The corresponding research question has a focus on usability and investigates when so called *reminders* are closed.

Every element in a workflow can be augmented by such a reminder element. Elements can be e.g. tasks, repetitions, decision questions. It can be defined after which time interval the participant is reminded of an element. This is especially helpful if elements require a longer time or participants are often distracted. It is interesting to see when reminders are closed, directly when the reminder appears on the screen or when the task is finished, and to see how often the reminders appear.

The idea is that if a user confirms the reminder more often per element, this indicates that the user is more tightly interacting with the device, while less often (e.g. only once per task) may indicate that the user rather interacts with the device only after completing a task (*m10* collects additional information for the question). However, since the number of confirmed reminders per task needs to be known before computing the average, These are counted per task (*m11_private*), but are stored only temporarily for the calculation.

The average can then be recalculated over the temporary values every time a value changes (*m11*). Both metrics are recorded at the same record points: when a (task-)block is entered, which initializes the count to 0, and every time a reminder is accepted, which increases the count by one (*Option* `constValueMap`). Since (*m11_private*) is counted for every task of an execution per workflow separately, its assignment is (w,e,t). Meta metrics are recorded after other metric types have been computed, therefore after counting, we always compute the latest average over all temporary values (*m11* is referenced by `metaReference`). After the workflow is completed or aborted, all temporary snapshot values collected by *m11_private* for this workflow can be deleted (`deleteSnapshots`).

Another example is metric *m13*, where the number of actively canceled schedules per week is counted. This results in a separate value for each week, which can be used later on to gain insight about the change over time (aggregate `time`). Since the value is not tied to a specific schedule, workflow etc., it is collected per user resp. device (global) in the case of shared devices. Every time a `scheduleAbort` event occurs, our count (the `sum`) for this week is increased by one (`constValue`). The number of snapshots, i.e. in this case the number of retained weeks where at least one schedule was aborted, is not limited in the study context.

It is also possible to record values as a parameter of the record point, as it is done e.g. in *m17* for recording the average setting of the text-to-speech speed.

5 METRIC ARCHITECTURE & METRIC LANGUAGE

In the following a domain specific language (DSL) for describing metrics and the corresponding metric architecture for the integration of metrics in an application are described (Figure 3).

5.1 Metric Language

In order to allow a flexible addition and reconfiguration of metrics, a DSL for describing metrics was defined based on requirements identified during the development of metrics in section 4. This DSL is descriptive rather than containing code for evaluating the metrics. The idea behind this was to simplify the specification and readability of metrics, reduce code duplication, and to integrate the consideration of privacy-related questions into the DSL, e.g. “What data is needed? Which time resp. value accuracy? Are timestamps needed for each measurement?”. Each time new research questions are added or research questions are modified and hence metrics need to be added or adjusted this should be done in a privacy-friendly way.

Additionally this specification of data capturing allows to derive automatically information about which data is collected, how this is done and when the data is collected as input for a privacy dashboard (Murmann and Fischer-Hübner, 2017) to foster transparency for users. The DSL is defined based on

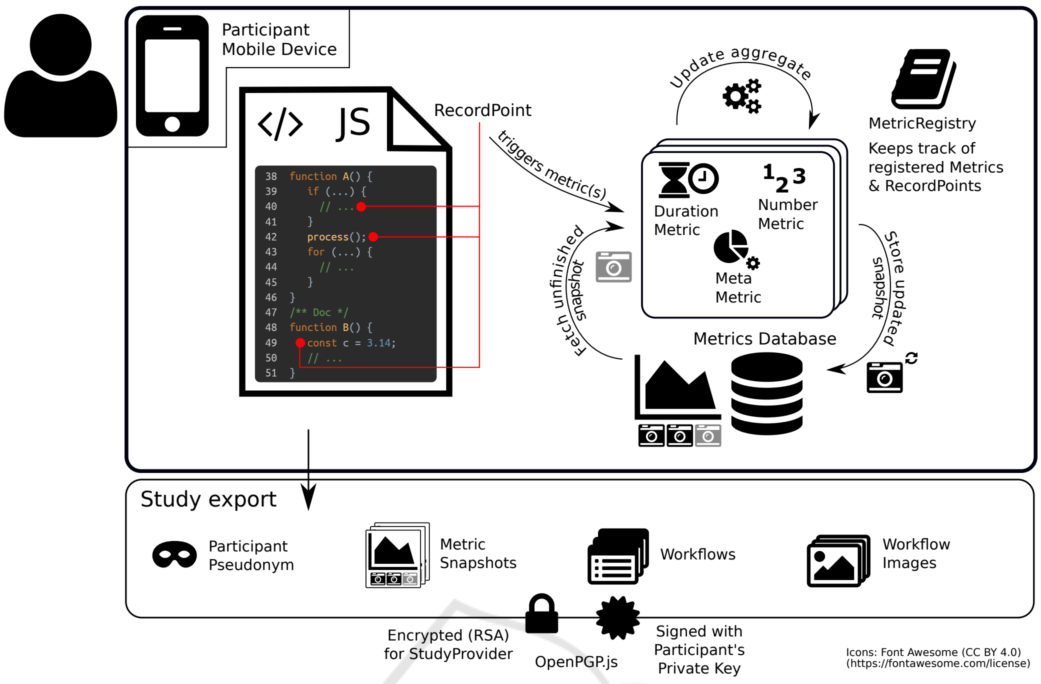
Table 3: Excerpt of declared metrics and their properties.

#	Subject	Type	Assign.	Record point	Aggregate		#Sn	Options
					op.	time		
m1	Number of workflow executions per workflow	int	w	workflowStart	sum	all	1	constValue: 1
m3	Number of completed workflow executions per workflow	int	w	workflowEnd	sum	all	1	constValue: 1
m8	Time taken per (actively) canceled workflow execution	dur	w,e	workflowStart →workflowAbort	–	–	∞	durationAccuracy: 1s, clearIncomplete: [workflowEnd, workflowStart], handleIncomplete: ignore
m10	Average time between confirmation of last reminder and marking a task as finished per execution of a workflow	dur	w,e	reminderAccept →blockAccept_ withAccepted Reminder	avg	all	1	durationAccuracy: 1s, clearIncomplete: [workflowEnd, workflowAbort], handleIncomplete: ignore, agg.durationAccuracy: 1s
m11 _private	Number of confirmed reminders per task per execution of a workflow	int	w,e,t	reminderAccept, blockEnter	sum	all	∞	private: true, constValueMap: {reminderAccept: 1, blockEnter: 0}, deleteSnapshots: [workflowEnd, workflowAbort]
m11	Average number of confirmed reminders per task per execution of a workflow	meta	w,e	reminderAccept, blockEnter	avg	all	1	metaReference: m11_private
m13	Number of (actively) canceled schedules per week	int	global	scheduleAbort	sum	week	∞	constValue: 1
m17	Average speed setting of TTS in workflow executions	float	global	workflowStart_ ttsSpeed(x)	avg	all	1	

JavaScript Object Notation (JSON), as it is a common, human-readable format already widely used in web and mobile applications. The language is specified as a TypeScript definition file and therefore allows syntax checking and auto completion in integrated development environments (IDEs). Furthermore many possible mistakes in the description of a metric can be detected at compile-time through rather restrictive types.

To separate the actual raw measurements and events from the metrics depending on them and to keep the actual metric binding code less intrusive (i.e. when measurements happen in application code), the concept of *record points* was introduced. These are places in the application code, where a potentially metric-relevant event happens, possibly including a context-related value. In the description of a metric, such record points may be referenced via their name, which causes the metric to be bound to that particular record point. Every time the event occurs, all metrics with a binding to the record point will be executed and will record a value. This decouples the definition of metrics from the events in code, i.e. instead of specifying which metrics are relevant at a certain event, only the event is specified and metrics declare what events they listen to. New metrics, which reuse existing record points, can be added simply by adding a definition, without changing further code.

Metrics differ in the *types of values* they record and their *recording behavior*. The simplest types are integer (`int`) and floating point (`float`) metrics, which record values of the particular type. In order to prevent type confusion, i.e. recording floating point values in an integer metric, runtime checks are performed. If no value is provided at the record point, the metric has to either specify a constant value for all record points (*Option* `constValue`) or for each record point separately (*Option* `constValueMap`). For example if only the count of certain events is needed, a constant value may be sufficient. Metrics can also record a duration. A duration is specified by two record points – a start (`recordStart`) and a stop (`recordStop`) – and the duration is measured between both events. The last type are meta metrics, which can compute new values derived from a base metric referenced through its name in `metaReference`. Metrics can be marked as `private` which causes them to be excluded from ex-


```
function A() {
   if (...) {
     // ...
   }
   process();
   for (...) {
     // ...
   }
}
/** Doc */
function B() {
   const c = 3.14;
   // ...
}
```

Participant Mobile Device
JS
RecordPoint
triggers metric(s)
Update aggregate
MetricRegistry
Keeps track of registered Metrics & RecordPoints
Duration Metric
Number Metric
Meta Metric
Fetch unfinished snapshot
Store updated snapshot
Metrics Database
Study export
Participant Pseudonym
Metric Snapshots
Workflows
Workflow Images
Encrypted (RSA) for StudyProvider
OpenPGP.js
Signed with Participant's Private Key
Icons: Font Awesome (CC BY 4.0) (https://fontawesome.com/license)

Figure 3: Metric Architecture.

ports and therefore makes them inaccessible for the study provider. This can be useful in combination with meta metrics, which are based on private metrics and compute an aggregate value. The separation into private and public metrics leads to further data minimization even for aggregate functions such as the median, which require storing all (counts of) values.

All metric types have in common, that they have an *assignment*, which states the scope in which metric values are collected or aggregated. This enables not only to record global metrics, but also metrics which are relating to a certain context. An example would be *m3* which counts the number of completed workflow executions *per workflow*. In this example the assignment is to the workflow, since the value is needed for each distinct workflow. Assignments can also be empty (for global metrics) or consist of multiple elements. For example to measure the time per completed execution per workflow (`assignment: ["workflow", "execution"]`). Specifying only `"execution"` would result in in an aggregated value for executions of workflows in general, while specifying only `"workflow"` would not group the recorded values per execution. This is especially relevant when computing aggregates, which are computed for each group separately.

A central aim of the DSL for metrics is to aggregate values on the device instead of collecting raw data. For *aggregation* operations the metric description has to include the otherwise optional `agg` element and specify the aggregate `operation`, as well as the `time` frame over which the aggregation should be performed. Currently supported aggregate operations are `min`, `max`, `sum`, `average`, `median` and `variance`, while supported time frames are `week`, `day`, `1/4day`, `hour`. Furthermore there is the special time frame `all`, which states that the aggregation should be over the complete time while metrics are recorded. Aggregation operations are performed in an incremental fashion, if possible, i.e. temporary values are kept to a minimum. For example we use Welford's online algorithm (Welford, 1962; Chan et al., 1983) for computing the variance in a numerically stable way.

Recorded values are stored in so-called *snapshots*. Every snapshot additionally has an index, the concrete assignment (e.g. workflow identifier and execution index) and may optionally store additional information for tracking aggregate values (such as welford state, number of measurements) and the start of the measurement (rounded timestamp). To limit the amount of recorded data, metrics have to specify the maximum amount of snapshots, which might also be infinite. This for example allows to keep data for a fixed number of weeks when combined with the time frame `week` for aggregation, or a fixed number of the last recorded values. The oldest snapshot is deleted, when a new one is to be recorded when the maximum number of snapshots is reached.

Metrics may also specify *accuracy* of values or duration. This can also be used to built histograms, since values in a certain range fall into the same accuracy bucket. Accuracy is described by a number and an optional unit in the case of duration accuracy. For example `[5, "s"]` states that all durations should be only stored to an accuracy of five seconds. An accuracy can be applied to raw values and aggregate values.

Often it is not necessary to store the *timestamp* of an event, as it may be sufficient to store the order of snapshots. Sometimes even that might not be required. Also when a timestamp is necessary, in almost all cases it does not need to have the full accuracy provided by the system. For the use case investigated here as timestamp in certain metrics the hour of events is recorded. Metrics may declare that a timestamp should be recorded, by specifying its accuracy in the `timestamp` field. Timestamps are also automatically recorded for aggregate metrics with a `time` frame option which is not `'all'`.

Since private metrics may accumulate many snapshots, which may not be necessary after an aggregation operation is finished there is an additional option (`deleteSnapshotsEvents`) to delete snapshots of a metric on certain record points or events.

In addition it needs to be defined what should be done with accumulated snapshots of private metrics in exceptional situations. There are two cases considered in the DSL: When the next duration measurement of the same metric and assignment starts, but the previous has not finished yet, it has to be decided how the incomplete measurement should be handled (Option `handleIncomplete`). It could be decided to `ignore` the measurement, or to `truncate` the duration to end at the current time, i.e. when the next measurement starts. If a certain measurement cannot be completed and there is no chance that another measurement with the same assignment would occur e.g. since the defined end of the study is reached, the snapshots which are accumulated for these measurements have to be deleted (*Option* `clearIncompleteEvents`) An example of a metric definition is shown in Listing 5.1.

Listing 1: Example metric definition.

```
{
    "name": "Number of workflow executions per workflow per
           week",
    "type": "int",
    "recordPoints": ["workflowStart"],
    "constValue": 1,
    "assignment": ["workflow"],
    "snapshots": "inf",
    "agg": {
      "operation": "sum",
      "time": "week"}
}
```

5.2 Metric Architecture

For the integration of the metric DSL the following metric architecture was developed. The code responsible for creating a record point by coupling productive code to the metric architecture, should be as minimal as possible. Furthermore logic regarding metrics should be kept at a minimum by describing most of the logic in the metric definition. In the metric architecture presented here the coupling can be realized by two different methods `record`, `recordValue`. which accept as parameters the record point name, the concrete assignment (e.g. workflow and execution identifier), as well as a value (`float` or `int`) in the case of `recordValue`.

Metrics are registered by providing a metric definition to a `MetricRegistry`, which keeps track of all record points and associated metrics. Record points are embedded in the code by calling the `record` or `recordValue` methods of the `MetricService`. This allows the flexible addition or modification of metrics, as definitions can be changed in a descriptive manner independent of their record points. Furthermore record points can mainly be added with the addition of a single line in the production code (Listing 5.2) since most logic, such as aggregation, is described in the metric definition.

Metrics are only recorded until the end of the study (predefined date). An export of the study data (Figure 3) is then encrypted with the public key of the study provider and signed with the participant's private key using OpenPGP.js[5].

Listing 2: Code changes necessary for adding a new metric and record point.

```
// register metric (application startup)
const metricDefinition = ...; //metric definition in DSL
metricService.register(metricDefinition);

// ... Somewhere in production code ...
// assignment: contains workflowId, executionId etc.
metricService.record('exampleRecordPoint', assignment);
// or: (value is a number)
metricService.recordValue('rpValue', assignment, value);
```

6 PRELIMINARY RESULTS

At the moment, the presented metric concept is evaluated in the context of several studies. First preliminary results are presented here. A group of $n = 4$ participants executed workflows while metrics were recorded for a period of five weeks. The results of the full study will be published in a separate publication. Three different types of workflows were modeled by

[5]https://openpgpjs.org/

Table 4: Preliminary results: participant number, workflow, number of started, completed workflows and average execution duration.

# p.	workflow	# start.	# comp.	Ø dur.
1	Shopping	11	10	32.11 s
2	Clean-up	8	4	6290 s
3	Public transport	9	9	38.11 s
4	Clean-up	4	3	5659 s

neuropsychologists in the team: shopping, apartment cleaning and using public transport.

In the following results of the preliminary evaluation are summarized. The number of started workflow executions per patient was relatively low ranging between four and eleven executions. The number of finished workflows (not canceled) had a lower range between three and ten. The total number of finished workflow executions over all patients was 26. Half of the participants had a workflow execution duration which fits the workflow type, while other executions indicate that some have been completed in less time than we would expect for that task. This may indicate that workflows have been skipped through while not performing the tasks during that time. For example the participants may have looked through the workflow before actually performing the tasks. Half of the participants had mostly workflows with relatively short execution times compared to the workflow type.

Mainly two types of errors occurred: one workflow was running for multiple days which is not plausible for the workflow type, 6 times workflows were canceled not inside the mobile application, where this would also be possible, but by external causes. Possible causes are that the app is terminated by the user or the operating system or the device lost power. Table 4 shows the workflows for each participant together with the number of started and completed executions, and the average execution duration. Errors are excluded from the statistics in Table 4.

7 CONCLUSION

Collecting data during empirical studies with mobile devices gives important insights beside pre/post tests. The approach presented here shows that it is possible to derive metrics from research questions ensuring data minimization via aggregation and decentralization in the context of a Privacy by Design approach. The presented concept is promising. The metric language and the metric architecture ensure that the approach is flexible and research questions can easily be described and adjusted. It is intended to use the metric language also as a basis for automatically generated privacy dashboards to inform users about data processing in a transparent way.

ACKNOWLEDGEMENTS

This work was supported by the Ministry for Science and Culture of Lower Saxony as part of SecuRIn (VWZN3224).

REFERENCES

Beierle, F., Tran, V. T., Allemand, M., Neff, P., Schlee, W., Probst, T., Zimmermann, J., and Pryss, R. (2019). What data are smartphone users willing to share with researchers? *Journal of Ambient Intelligence and Humanized Computing*, pages 1–13.

Bertens, D., Fasotti, L., Boelen, D. H., and Kessels, R. P. (2013). A randomized controlled trial on errorless learning in goal management training: Study rationale and protocol. *BMC Neurology*, 13(1):64.

Bonawitz, K., Ivanov, V., Kreuter, B., Marcedone, A., McMahan, H. B., Patel, S., Ramage, D., Segal, A., and Seth, K. (2017). Practical secure aggregation for privacy preserving machine learning. Cryptology ePrint Archive, Report 2017/281. https://eprint.iacr.org/2017/281.

Cafazzo, J. A., Casselman, M., Hamming, N., Katzman, D. K., and Palmert, M. R. (2012). Design of an mHealth App for the Self-management of Adolescent Type 1 Diabetes: A Pilot Study. *Journal of Medical Internet Research*, 14(3):e70.

Chan, T. F., Golub, G. H., and Leveque, R. J. (1983). Algorithms for computing the sample variance: Analysis and recommendations. *The American Statistician*, 37(3):242–247.

Colesky, M., Hoepman, J. H., and Hillen, C. (2016). A critical analysis of privacy design strategies. In *2016 IEEE Security and Privacy Workshops (SPW)*, pages 33–40.

Di Matteo, D., Fine, A., Fotinos, K., Rose, J., and Katzman, M. (2018). Patient willingness to consent to mobile phone data collection for mental health apps: structured questionnaire. *JMIR mental health*, 5(3):e56.

Eckhardt, K., Schiering, I., Gabel, A., Ertas, F., and Müller, S. V. (2019). Visual Programming for Assistive Technologies in Rehabilitation and Social Inclusion of People with Intellectual Disabilities. In *MuC ’19*.

Emmanouel, A. (2017). *Look at the frontal side of life: Anterior brain pathology and everyday executive function: Assessment approaches and treatment*. PhD thesis, Radboud University.

European Union (2016). Regulation (EU) 2016/679 of the european parliament and of the council of 27 april

2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing directive 95/46/EC (general data protection regulation). *Official Journal of the European Union*, L119:1–88.

Gabel, A., Schiering, I., Müller, S. V., and Ertas, F. (2018). mHealth Applications for Goal Management Training - Privacy Engineering in Neuropsychological Studies. In Hansen, M., Kosta, E., Nai-Fovino, I., and Fischer-Hübner, S., editors, *Privacy and Identity Management. The Smart Revolution: 12th IFIP WG 9.2, 9.5, 9.6/11.7, 11.6/SIG 9.2.2 International Summer School, Ispra, Italy, September 4-8, 2017, Revised Selected Papers*, IFIP Advances in Information and Communication Technology, pages 330–345. Springer International Publishing, Cham.

Garcia-Ceja, E., Osmani, V., and Mayora, O. (2016). Automatic Stress Detection in Working Environments From Smartphones' Accelerometer Data: A First Step. *IEEE Journal of Biomedical and Health Informatics*, 20(4):1053–1060.

Goyal, R., Dragoni, N., and Spognardi, A. (2016). Mind the Tracker You Wear: A Security Analysis of Wearable Health Trackers. In *Proceedings of the 31st Annual ACM Symposium on Applied Computing*, SAC '16, pages 131–136, New York, NY, USA. ACM.

Grünerbl, A., Muaremi, A., Osmani, V., Bahle, G., Öhler, S., Tröster, G., Mayora, O., Haring, C., and Lukowicz, P. (2015). Smartphone-Based Recognition of States and State Changes in Bipolar Disorder Patients. *IEEE Journal of Biomedical and Health Informatics*, 19(1):140–148.

Hansen, M., Jensen, M., and Rost, M. (2015). Protection goals for privacy engineering. In *2015 IEEE Security and Privacy Workshops*, pages 159–166.

He, D., Naveed, M., Gunter, C. A., and Nahrstedt, K. (2014). Security Concerns in Android mHealth Apps. *AMIA Annual Symposium Proceedings*, 2014:645–654.

Huckvale, K., Prieto, J. T., Tilney, M., Benghozi, P.-J., and Car, J. (2015). Unaddressed privacy risks in accredited health and wellness apps: A cross-sectional systematic assessment. *BMC Medicine*, 13:214.

Jamieson, M., Jack, R., O'Neill, B., Cullen, B., Lennon, M., Brewster, S., and Evans, J. (2019). Technology to encourage meaningful activities following brain injury. *Disability and Rehabilitation: Assistive Technology*, 0(0):1–14.

Kleiman, E. M., Turner, B. J., Fedor, S., Beale, E. E., Picard, R. W., Huffman, J. C., and Nock, M. K. (2018). Digital phenotyping of suicidal thoughts. https://onlinelibrary.wiley.com/doi/abs/10.1002/da.22730.

Knorr, K. and Aspinall, D. (2015). Security testing for Android mHealth apps. In *2015 IEEE Eighth International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, pages 1–8.

Levine, B., Robertson, I. H., Clare, L., Carter, G., Hong, J., Wilson, B. A., Duncan, J., and Stuss, D. T. (2000). Rehabilitation of executive functioning: An experimental–clinical validation of goal management training. *Journal of the International Neuropsychological Society*, 6(3):299–312.

Martínez-Pérez, B., de la Torre-Díez, I., and López-Coronado, M. (2015). Privacy and Security in Mobile Health Apps: A Review and Recommendations. *Journal of Medical Systems*, 39(1):181.

McMahan, B., Ramage, D., Talwar, K., and Zhang, L. (2018). Learning differentially private recurrent language models. In *International Conference on Learning Representations (ICLR)*.

Mense, A., Steger, S., Sulek, M., Jukic-Sunaric, D., and Mészáros, A. (2016). Analyzing privacy risks of mHealth applications. *Stud Health Technol Inform*, 221:41–45.

Müller, S. V., Ertas, F., Aust, J., Gabel, A., and Schiering, I. (2019). Kann eine mobile anwendung helfen abzuwaschen? *Zeitschrift für Neuropsychologie*, 30(2):123–131.

Mohr, D. L., Zhang, M., and Schueller, S. M. (2017). Personal Sensing: Understanding Mental Health Using Ubiquitous Sensors and Machine Learning. *Annual review of clinical psychology*, 13:23–47.

Morera, E. P., Díez, I. d. l. T., Garcia-Zapirain, B., López-Coronado, M., and Arambarri, J. (2016). Security Recommendations for mHealth Apps: Elaboration of a Developer's Guide. *Journal of Medical Systems*, 40(6):152.

Murmann, P. and Fischer-Hübner, S. (2017). Tools for achieving usable ex post transparency: a survey. *IEEE Access*, 5:22965–22991.

Onnela, J.-P. and Rauch, S. L. (2016). Harnessing Smartphone-Based Digital Phenotyping to Enhance Behavioral and Mental Health. *Neuropsychopharmacology*, 41(7):1691–1696.

Papageorgiou, A., Strigkos, M., Politou, E., Alepis, E., Solanas, A., and Patsakis, C. (2018). Security and Privacy Analysis of Mobile Health Applications: The Alarming State of Practice. *IEEE Access*, 6:9390–9403.

Stamenova, V. and Levine, B. (2018). Effectiveness of goal management training® in improving executive functions: A meta-analysis. *Neuropsychological Rehabilitation*, 0(0):1–31.

Treacy, C. and McCaffery, F. (2016). Data Security Overview for Medical Mobile Apps. *International Journal on Advances in Security Volume 9, Number 3 & 4, 2016*.

Troncoso, C., Isaakidis, M., Danezis, G., and Halpin, H. (2017). Systematizing decentralization and privacy: Lessons from 15 years of research and deployments. *Proceedings on Privacy Enhancing Technologies*, 2017(4):404–426.

Welford, B. P. (1962). Note on a Method for Calculating Corrected Sums of Squares and Products. *Technometrics*, 4(3):419–420.