Data Accountability in Socio-Technical Systems

Kristian Beckers, Jörg Landthaler, Florian Matthes, Alexander Pretschner, and Bernhard Waltl

Technical University of Munich, Faculty of Informatics
kristian.beckers@tum.de,joerg.landthaler@tum.de,matthes@tum.de,alexander.pretschner@tum.de,b.waltl@tum.de

Abstract. Data-accountability encompasses responsibility for data and the traceability of data flows. This is becoming increasingly important for Socio-Technical Systems (STS). Determining root causes for unwanted events after their occurrence is often not possible, e.g. because of missing logs. A better traceability of root causes can be supported by the integration of accountability mechanisms at design time.
We contribute a structured method for designing an accountability architecture for STS at design time. Therefore, we propose the elicitation of accountability goals to answer why an unwanted event happened and who is responsible for it. We also identify four different interaction types in STS. Additionally, we derive accountability graphs from a generic accountability model for STS that serve as a baseline for designing accountability mechanisms for all relevant entities in an STS. The resulting architecture is adjusted to legal requirements, regulations and contracts. We demonstrate the applicability of our approach with an eHealth case study.

Key words: Data Accountability, eHealth, Socio-Technical Systems, Accountability Architecture, Interaction Types, Accountability Method, Accountability Graph

1 Introduction

An important aspect of information systems (IS) is compliance to legal standards and organizational policies. In case of a violation of a statute it becomes more and more important to identify responsible parties, i.e. to hold someone accountable, assuming that a system can not be responsible on it's own. Hence, an IS is embedded in a so-called Socio-Technical System (STS, (c.f. [1])) encompassing juristic persons and technical systems. Currently, the impact of technical systems to prior analogue world definitions of accountability is unclear. Only few attempts have been made to include the ability of tracing root causes of unwanted events by design, such as [2]. Current efforts for data accountability IS lack either the social aspect of IS or a solution by design or both. Feigenbaum et al. [3] adopts the concept of accountability that is known well in the analogue world to ensure security in information systems and proposes that the ability to punish people will prevent them from doing illegal actions. The question, how to

enable IS with this ability remains open. One much-noticed approach is proposed by Weitzner et al. [4] by demanding that each subsystem should be responsible to ensure accountability on its own using an appropriate accountability mechanism, e.g. by policy-aware transaction logs.

However, it is unclear how the interaction between humans and machines affect accountability. Within an organization it is important to identify (and define) responsible roles. Another important task is to identify relevant regulations and policies. We analyze possible interaction types in STS, use Data Governance principles to allocate responsibilities and provide a model of STS. Afterwards, we propose a structured method empowering engineers to enhance an STS with accountability by design, which can *a posteriori* determine causalities within such a system. Accountability in this work is a capability of an STS to answer questions regarding the cause of occurred unwanted events (e.g. privacy or security violations).

We limit ourselves to data accountability, i.e. unwanted events whose causes are data-related. Consider e.g. software malfunctions producing wrong data, hardware failures due to faulty interpretation of data or wrong usage of technical systems (intended or not) caused by wrong instructions. The system should be enabled to determine the causality and identify responsible parties. Accountability mechanisms (which could be, but is not restricted to, logging) need to answer the question why an unwanted event happened.

The creation of an accountability solution is related to legal compliance in two ways. Firstly, an accountability solution can support legal compliance. Companies have to ensure transparency and provenance of their data, e.g. when adherence to regulatory frameworks such as HIPAA in the U.S., [5], demands specific needs for the confidentiality and security of healthcare information that describe specific principles regarding transparency and provenance of data. Secondly, the introduction of an accountability solution itself can cause additional legal compliance demands. For example, the gathering and storage of personal data requires compliance to data protection acts, such as the EU Data Protection Directive (95/46/EC). Our structured method explicitly identifies and considers legal compliance as part of designing an accountability solution.

This paper is structured as follows: Section 2 reviews related work on data governance and accountability. Section 3 describes our structured method for creating an accountability architecture, Section 4 illustrates our accountability STS model, and in Section 5, we inherit STS relation types. We apply our method in Section 6 and conclude our research in Section 7.

2 Background and Related Work

2.1 Data Governance

Weill and Ross [6] presented a framework and structuring mechanism for IT Governance. For a management perspective, they identified five key decision domains: IT architecture, IT Infrastructure, IT investment and prioritization

decisions and business applications needs. Based on this research Khatri and Brown [7] focused on Data Governance and identified five decision domains, namely data principles, data quality, data access, metadata and data lifecycle. The authors use these domains as a blueprint for identifying and assigning data responsibilities, i.e. roles. However, their work does not include the causal aspect of accountability nor the relations between different roles. Our method shows how to use their framework to identify relations between roles to determine responsible persons for an unwanted event.

An important part of data governance, is to determine the origin of a certain datum by means of its source, often called Data Provenance, e.g. Buneman et al. [8]. Moreover, the manipulation history of a datum and potentially the person or other technical components or even a chain of components that led to the manipulation of the datum can be useful to answer accountability questions. We propose to design an accountability solution for a STS that allows to address challenges of Data Provenance.

2.2 Accountability

Weitzner et al. [4] proposed an understanding of accountability such that it reflects the ability of a system to answer questions regarding the why of occurred events. For example, why was personal data released to unauthorized staff?

Accountability is the subject of active research in different areas of computer science such as network engineering, see Bechtold and Perrig [9]. Fundamental for accountability is an understanding of causality in general (c.f. Gössler and Le Métayer [10], Halpern and Pearl [11]). In our research, it is essential to consider the efforts of Data Governance as explained above, which provide the essential information of responsible parties and rules of how data should be treated. In particular, effective Data Provenance is necessary to enable the traceability of data through an STS. Without these traces, answering the accountability question is impossible. A straight forward accountability mechanism is the employment of logs at every node of a system, which follows Weitzner et al. [4] . The authors proposed policy-aware transaction logs, which are created by logging of relevant information by individual entities in a system. Identifying fundamental accountability concepts of who and how are discussed in Eriksn [12].

We illustrate our approach using an eHealth example. Gajanayake et. al [13, 14] address a similar problem by applying information accountability to systems in the eHealth domain. However, they neither provide a structured method to design a general accountability architecture, nor incorporate data governance structures, nor examine causality chains.

3 A Structured Method for Designing an Accountability Architecture

We contribute a structured method for the design of an accountability architecture (see Fig. 1). Our method is presented in a sequential fashion for simplicity's sake; iterations between different steps are possible during its application.

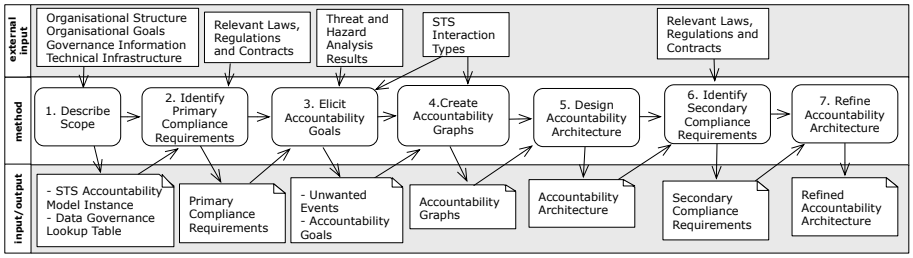

Fig. 1: A Method for Designing an Accountability Architecture

Step 1. Describe Scope Initially we have to understand the STS for which we want to elicit accountability. We propose an accountability model that contains abstract descriptions of all technical components and roles in the scope of an accountability system. Thereby, we describe interactions between these STS elements in order to be able to trace unwanted events to their sources in later steps of our method. We describe in detail how to model the STS and its relation in Sects. 4 and 5.

Step 2. Identify Primary Compliance Requirements We identify relevant compliance requirements arising from regulative texts for the STS as motivation for designing an accountability architecture. For example, if we process personal information in our STS in Germany the Federal Data Protection Act (BDSG) is relevant.

Step 3. Elicit Accountability Goals We elicit unwanted events by considering the STS model and the organizational goals of the customer. Note that in our work an unwanted event is any occurrence within an STS that concerns violations of safety, security, or privacy requirements. We distinguish unwanted events regarding these software qualities as follows. Safety analysis focuses on hazards caused by the engineers of soft- and hardware and random faults in these systems. Hazards are situations that lead to accidents that harm humans. Hence, in safety unwanted events are accidents. Security is about protecting an asset, an item of value for a stakeholder from threats caused by malicious attackers or unintentional acts of stakeholders. Realized threats are attacks. Thus, in security analysis unwanted events are successful attacks. Privacy concerns the protection of personal information of stakeholders. An unwanted event in privacy is a data leak of personal information to unauthorized stakeholders. Our work does not restrict the techniques for hazard or threat analysis. For space reasons, we exclude these analyses in this paper. Afterwards each unwanted event is mapped

to an accountability goal that describes the abilities the system shall have to identify causes for this particular unwanted event (see examples in Sects. 6). After having identified the cause for the unwanted events in the STS, we need to derive the responsible actor for that element. The information can be derived from existing governance data of the organization. For this purpose, we create a Data Governance Lookup Table mapping STS elements to responsible actors.

Step 4. Create Accountability Graphs We provide a divide and conquer approach for accountability to reduce the complexity of the overall design problem. For each accountability goal, we create a separate accountability graph as follows. Nodes in our graph are STS elements e.g. humans or machines, while edges are communication channels between these STS elements. The first node of the graph is the STS element where the unwanted event occurs. Afterwards we include all STS elements and relations that are part of the normal operations of the first node of the graph. Using the accountability graph, we have the capability to identify the potential loci of the root cause of the unwanted event by conducting a search along all relevant nodes of the information flow between involved entities. The reasoning for creating the graph is documented for later analysis and should be checked by independent experts, which shall prevent that our accountability graphs have incomplete information.

Step 5. Design Accountability Architecture We need to ensure that all elements in the accountability graph have the capability to support the information needed for our causal reasoning from the unwanted event to its source. In particular, each node has to have a mechanism to monitor and document relevant events. The result is an accountability architecture that ensures the satisfaction of the accountability goal.

Step 6. Identify Secondary Compliance Requirements we identify relevant secondary compliance requirements that can be identified for our resulting accountability architecture. For example, an intensive logging of personal information may conflict with a given privacy legislation. This step results in a set of compliance documents that are refined into precise compliance requirements for our proposed accountability architecture.

Step 7. Refine Accountability Architecture The planned accountability architecture is revised according to the compliance requirements. The result of this step is a compliant and precise description of an accountability architecture that satisfies the initial accountability goals, as well as the elicited compliance requirements.

4 A Generic Model for Socio-Technical Systems

As already stated, socio technical systems become increasingly complex. This has several reasons and up to certain extent this is due to the contained entities and their tight interconnectedness among each other. Results from Enterprise Architecture Management (EAM) have significantly improved the understanding of business, their capabilities and the interconnectedness to technical and physical entities Lankhorst [15], Jonkers et. al [16]. Based on the insights gained

from EAM, we are able to transfer those results into a generic STS model. The interconnectedness of the STS elements provides insights into the different communication channels between those elements, which we show in our generic STS model (depicted in Fig. 2).

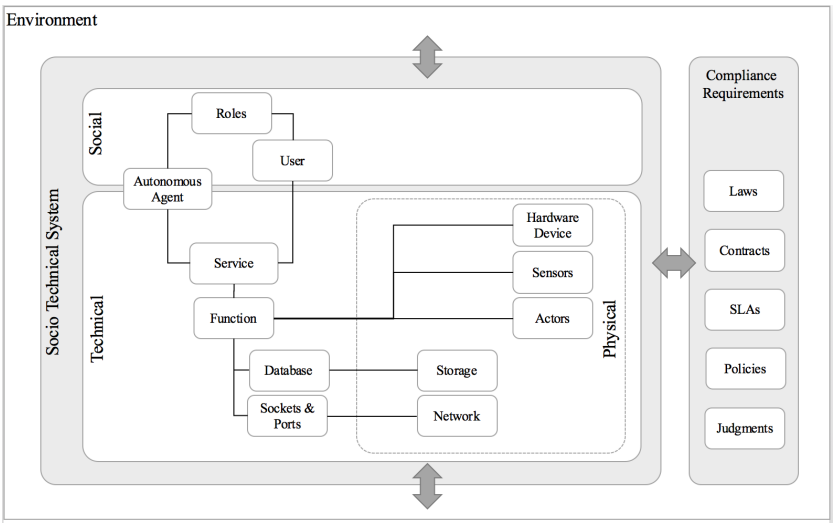

Fig. 2: A Generic Accountability Model for STS Differentiating Social and Technical Layers.

The model shown in Figure 2 consists of three different layers: social, technical and physical. Accountability in STS has several dimensions, which we are now going to examine. Thereby, we are able to assign each dimension to at least one of those three layers. Consequently, this approach provides a constructive and structured way of differentiating the term accountability into sub-problems, which can then be investigated separately. Trivially, the model shown in Figure 2 serves as a base line to answer the question regarding the dimensions of accountability; therefore, it remains - as every model - an abstraction from a real world STS. Nevertheless, it is comprehensive in the sense that we can use it to differentiate between the dimensions of accountability in interacting STS. Most relevant for the accountability dimensions are of course the obligations of an STS arising through legislation, contracts, SLAs and other policies governing the flow and management of data and information assets in general.

Social. The social dimension of an enterprise covers all organizational units and human actors, which are interacting among each other and with the technical systems. Commonly they are organized into roles aggregating them according to responsibilities, tasks and goals. In addition to the users, autonomously acting users, i.e. agents, get more and more in the focus of investigations regarding accountability.

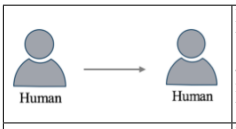	In order to reconstruct the behavior of a human, and the reasons for it, it might be necessary to understand and retrace the information (e.g. instructions) he got from another human. This reconstruction requires information which might not be codified properly (e.g. burden of proof), such that an explanation cannot be given.
	Human interact with machines, i.e. services, sensors, etc. provided by the technical and physical layer of the STS. However, if the human interacts with machines, such as insertion, update or deletion of information, this has effects on the technical layer. Consequently, to reconstruct the actions done by machines it is required to understand the triggers that caused the machine to perform a certain action.
	Machines can offer information to human, such as notifications about an event. The provision of information by a technical system causes the human to perform a follow-up action or hinders him from doing some actions. Keeping trace of the information that was offered to humans is not trivial, but essential in order to reconstruct the behavior of the STS.
	The interaction between machines, such as retrieving and aggregating data from sensor networks or forwarding commands to an actor that changes the physical environment, is the fourth interaction type in STS. Which service, respectively function, has processed which data and forwarded it to which instance, is a critical question that has to be answered by accountability mechanisms in STS.

Table 1: Interaction Types in STS between Humans and Machines

Technical and Physical. The technical dimension of an enterprise covers the application landscape with its services and functions. Hereby, "functions" can be understood in a technical sense, such that concrete functionalities, such as network communication and persisting data in databases, are subsumed. Those technical functions are aggregated to more complex services, which are later on consumed by users or agents to fulfil their needs. The physical layer is part of the technical layer and covers hardware devices and interactions at hardware level. Every device that is either measuring physical phenomena or states, i.e. sensors, or changing the physical environment, i.e. actors, belongs to the physical layer. This differentiation in layers briefly shows how the components of an STS are interacting among each other. These interactions have to be investigated to fully understand the challenges and drawbacks of data accountability (see Section 5).

5 Interaction Types in STS

Based on the model for an interacting STS (see Figure 2) we distinguish the different types of interactions. We identified four different types of interactions, namely human-human, human-machine, machine-human, machine-machine, which we describe in Table 1. Those interaction types have an impact on the design and implementation of the accountability mechanism.

Accountability mechanisms in STS have to consider these interaction types, otherwise no comprehensive reconstruction of behavior and explanations can be performed. This has consequences for the design of such accountability mechanisms. Those mechanisms heavily influence the way in which data has to be tracked and logged in the overall STS and how this data can be stored. Based on

this stored data it is possible to automatically derive accountability information and to reconstruct the root cause of an event, c.f. Waltl et al. [17].

6 Case Study

We illustrate our approach with the case study *eHealth Record (EHR)* adapted from the NESSoS[1] project. EHRs contain any information created by health care professionals or devices in the context of the care of a patient. Examples are laboratory reports, X-ray images, and data from monitoring equipment. The method will be executed by a dedicated accountability officer in coordination with lawyers, domain experts, and software engineers.

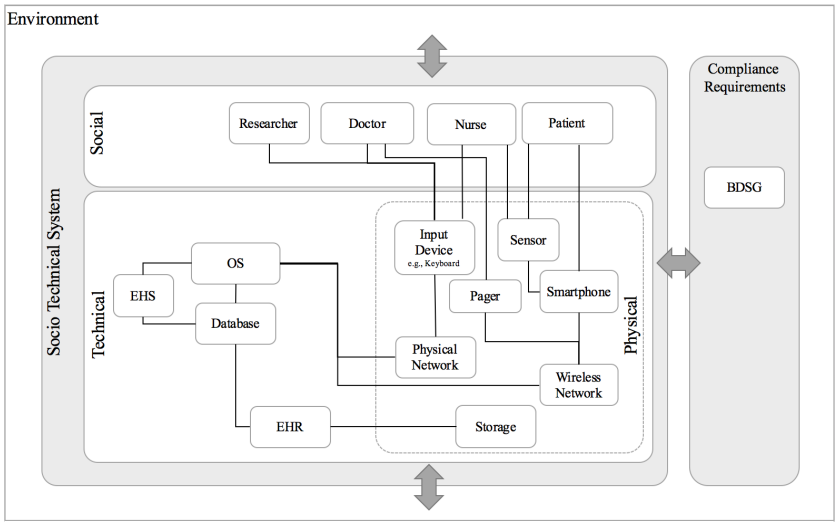

Fig. 3: Instance of our Accountability Model for STS for the eHealth Scenario

Step 1. Describe Scope EHRs are part of an eHealth System (EHS) owned by a hospital. The overall organizational goal of a hospital is to fulfil the societal goal to provide health care for patients. An EHS with its EHRs shall help to treat patients more efficiently and effectively. For example, the nurse does not need to take the vital signs for specific time intervals and deliver them to the doctor manually, because the EHS fulfils these tasks automatically, hence saving working time. We illustrate our example in Figure 3. The EHS is a software that stores medical information in EHRs. Further, it interacts with different users and communicates with various devices and serves as the example of an STS. In Germany, an EHS has to be compliant with the Federal Data Protection Act

[1] Network of Excellence on Engineering Secure Future Internet Software Services and Systems (NESSoS), http://www.nessos-project.eu, last access on 03/23/2016

Data Governance Responsibility	**Responsible Role**
(D1) Data in the Database	(R1) Data Owner (Doctor)
(D2) Update EHR data	(R2) Data Steward (Nurse)
(D3) Sensor maintenance	(R3) Data Custodian (Nurse)

Table 2: An Excerpt of a Data Governance Lookup Table

(BDSG). Hence, the information stored in the EHR shall only be accessed with the patient's informed consent. An exception to this rule is a medical emergency, in which case the patient's physical status may prevent her from giving the consent. In addition, the information in the EHR supports clinical research, which is represented by a researcher in this scenario. The patient wears a sensor that is monitoring her vital signs and communicates them to the patient's smartphone. The smartphone is transmitting the data via a wireless network to the EHS. Doctor, nurse and researcher use a terminal that is connected to the EHS vial a physical network. The doctor carries a pager in order to receive emergency calls from the EHS. The EHS is embedded in the organizational structure of a hospital. In the following, we design a runtime accountability mechanism for this example scenario. We start by defining the data governance roles for our EHR scenario. Hospitals often host a large IT landscape for various purposes. The EHS supports different user roles and is also embedded in the organizational structure of a hospital. A Data Owner is essential to our method. Note that we refer to a data owner in the sense of Data Governance according to Khatri and Brown [7]. A Data Owner is the trustee responsible for data and its uses. In our example the doctor is the Data Owner being in charge of the health data in general, be cause on the one hand he is allowed to access the data while on the other hand he often assumes managerial tasks regarding EHRs in a hospital. The nurse is assigned a Data Steward role that enters certain data about patients into the EHS. A Data Steward according to Khatri and Brown [7] is responsible for what is stored in a set of data. This is a delegation of responsibilities from the doctor as Data Owner to the nurse as Data Steward. Furthermore, the nurse is also assigned the Data Custodian role. A Data Custodian takes care of a working technical infrastructure for collecting or transporting data. In our case, the nurse has to take care of the maintenance of the sensor e.g. exchanging its batteries. The patient assumes the role of a Data Provider that repeatedly sends data from a sensor to the EHS where it is then stored. A typical Data Consumer is the researcher that merely receives data for his research activities. Next, we need to elicit the accountability goals. Table 2 shows an excerpt of a Data Governance Lookup Table.

Step 2. Identify Primary Compliance Requirements The German Federal Data Protection Act (BDSG)[2] is relevant for an application of the proposed system, because it processes the personal information of the patient. The § 3 of the BDSG states that personal information can only be elicited, stored and

[2] Note: We will address the inclusion of further laws and resolving conflicts between them in the future and focus in this paper exclusively on the BDSG.

Unwanted Events	Accountability Goals
(E1) Patient is in an emergency and does not get help from the doctor.	(G1) Ability to reconstruct the root cause for the patient not receiving immediate help from the doctor during the medical emergency.
(E2) Patient received wrong treatment from the nurse.	(G2) Ability to analyze why the patient received a harmful treatment from the nurse.
(E3) Researcher access PII from Patient: name, disease, vital signs	(G3) Ability to identify the cause of the data leak of the Patient's PII.

Table 3: Selected Unwanted Events and Accountability Goals

processed for a specific purpose and have to be anonymized if possible. In this case the doctor and nurse need to know the identity of the patient to be able to diagnose and administer treatment. Moreover, according to § 4 of the BDSG the patient has to provide an informed consent about the processing of her personal information and who will access it.

Step 3. Elicit Accountability Goals We show three prototypical unwanted events and their respective accountability goals in Table 3. Note that for space reasons we omit the threat and hazard elicitation. We focus in our example on events that physically harm the patient or violate the patient's privacy. We derive accountability goals for each event that demand an accountability mechanism to trace the causes for this particular unwanted event within the scope of the STS.

Step 4. Create Accountability Graphs We choose accountability goal (G1) *why the patient received a harmful treatment from the nurse* for the remainder of this example. We trace back all involved elements of the STS in the instance of the accountability model from the patient to the doctor and gather a sub graph of elements and their relationships. The resulting accountability graph is depicted in Figure 4. All of these elements can cause the missing communication of the patient's emergency to the doctor. The elements were selected based on the information in the hazard and threat analysis. In our example for this particular accountability goal, we do not consider the researcher, because a data leak does not relate to this specific accountability goal.

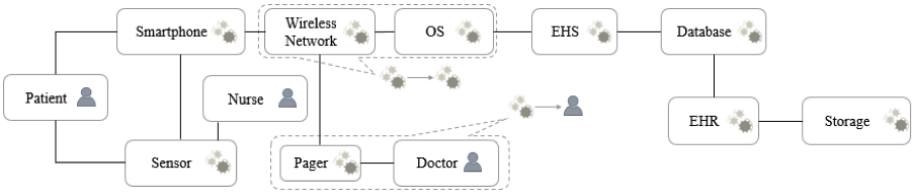

Fig. 4: Accountability Graph for an eHealth Scenario (2 interaction types shown)

Step 5. Design Accountability Architecture We design an accountability architecture, which is comprised of several local accountability mechanisms. For

each accountability goal all possible nodes and edges of the corresponding accountability graph need to be assessed for the demand of a separate accountability mechanism with respect to the accountability goal under consideration. The intention of this assessment is to find adequate parts of the accountability graphs so that individual causes can be localized with respect to the organizational needs. We are aware that there exist different types of accountability mechanisms, e.g. digital or analogue that are also potentially limited. We model each accountability mechanism as an STS, too. Following the policy-aware transaction log accountability mechanism, the patient needs to document every medicine ingestion. Both, analogue or digital logs, e.g. handwritten or via a tablet can be considered. A problem that arises in the design phase is the level of granularity that the information needs to have. The information should not be too detailed, because the identification of relevant details will take time and resources and the information should not be too abstract in order not to miss vital information for identifying the cause of the unwanted events.

Step 6. Identify Secondary Compliance Requirements Humans have the right for transparency according to §§ 19,34 of the BDSG for any system that processes their personal information. In particular, transparency demands a detailed and complete report on the life cycle of the personal information. An accountability architecture as proposed in this work has the ability to provide this information with little effort. The accountability graph provides an abstract view of the flow of personal information in the system, which can be accompanied with detailed access logs of all persons reading or changing the personal information of the patient. These data provenance capabilities of the chosen accountability mechanism will improve the transparency of STS significantly, because all the foundations for providing detailed reports to affected persons will be available. However, these large amounts of personal data of the patient have to be protected from access of further actors in this scenario. For example, doctors or nurses that are employed by the hospital but are not involved in the treatment of the patient have to be prevented from gaining access to that data (BDSG §9). Moreover, the data has to be deleted after the purpose for its initial collection is not valid anymore, e.g., the patient is no longer treated (BDSG §§ 20, 35).

Step 7. Refine Accountability Architecture We illustrate our resulting accountability architecture in Figure 5. The architecture is comprised of individual accountability mechanisms that ensure the logging and monitoring of individual components and delivering these logs to the accountability evaluation part of our architecture. The evaluation takes care of analyzing the log files and answering the why and who questions of accountability. The resulting compliance requirements of Step 6 are incorporated into the architecture. For example, we have to incorporate access control mechanism for the data and a process that checks if the purpose for storing the data has not expired. This needs to be done for all accountability mechanisms to ensure a holistic solution for these problems.

We analyze all involved components of our accountability graph in detail for our exemplary unwanted event *Patient is in an emergency and does not get help from the doctor* and determine which data of the component has to be stored

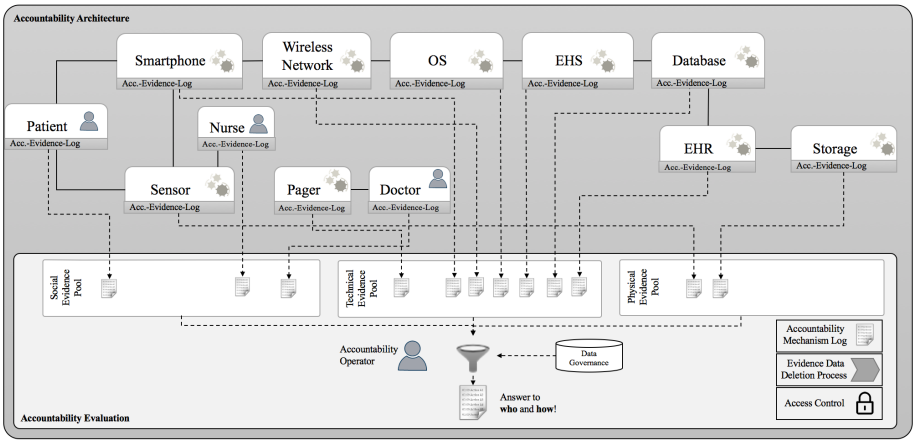

Fig. 5: Resulting Accountability Architecture for the eHealth Scenario

to be able to determine if this component was (part of) the root cause of the unwanted event. Additionally, for a policy-aware transaction log accountability mechanism for all components, it needs to be decided whether information that is forwarded to further components of the accountability graph needs to be stored in the component.

We consider in our example that the patient has a heart attack and the sensor monitoring his heart frequency should report this to the doctor. We choose the sensor as first component to consider in our accountability architecture. We need to log what information the sensor is capturing and at what time. The log can answer the first accountability question, did the sensor malfunction and did not record the correct heartbeat. We have to ensure that the log exists over time. Due to limitations of the sensor's memory capacity, the information has to be transported and stored on the smartphone. Each time a batch of information is transferred to the smartphone, the log file in the sensor stores a hash of the transported information and the date of transmission. This allows checking at the smartphone if all the data from the sensor has arrived at the smartphone. Moreover, the sensor transmits the heartbeat every 20 seconds to the smartphone. We have to implement a logging mechanism at the sensor that persists the information what was send to the smartphone at what time. This information allows us to decide if information was not send by the sensor or not received by the smartphone. Furthermore, we have to determine similar decisions for the smartphone, e.g., check that data was evaluated correctly and that the smartphone send an emergency message via the wireless network and the OS to the EHS and finally to the pager of the doctor.

So far all considered interaction types are of the machine-to-machine interaction type (see Section 5), which allowed us to specify automatic logging procedures. The doctor shares a human-machine interaction type with our system, which means that the she has to manually log her activities. For our example,

a policy states that the doctor has to log all his reactions to received pager message. The purpose of this policy is that in a post mortem analysis it can be decided if the doctor reacted to all pager messages reasonably. A sensor's digital log file can be limited by its memory size. However, the sensor's log file can be sent and aggregated on the smartphone and still fulfil the accountability goal with respect to the organizational needs. Either you detect a node that is a machine, hence there needs to be a lookup in the Data Governance Lookup Table (see Tab. 2) who is responsible for the machine, or a role is detected and a lookup can be necessary, too.

In our use case, we find e.g. anomalies in the sensor log aggregated on the smartphone. This happened due to missing replacement of batteries from the data custodian of the sensor that is of the type machine. This is accounted by a missing manual log entry of the data custodian. Hence, we need a lookup in the Data Governance Lookup Table in order to find the responsible person for the machine. In this case, the nurse (R3) is responsible data custodian for the sensor maintenance. In addition to that, one could lookup who in the organization is the data owner of the sensor data, which in our case is the doctor. Hence, he also has a partial responsibility for the resulting problems of the patient. Further criminal investigations have to rule out any other causes such as the batteries have been robbed.

7 Conclusion and Future Work

This paper structures the accountability concept in socio-technical-systems (STS) by differentiating the four different types of interactions, namely human-human, human-machine, machine-human and machine-machine. Our approach is restricted to accountability on data. Consequently, we exclusively consider and analyze the flow of information, i.e. data, during the possible interactions. The interaction types allow a structuring of the various forms of accountability, offering an analytical way of defining accountability mechanisms considering relevant requirements arising from laws, SLAs, contracts, etc. Based on these insights we propose a structured method for deriving an accountability solution that incorporates functionalities for answering the questions of why an unwanted event did happen and who is responsible. We rely on previous work for data governance to answer the responsibility question and work on data accountability for answering the why question. We illustrate our approach by a case study in the eHealth domain. This proof of concept shows the applicability of our approach and is the baseline for our next steps, which are a more detailed conceptualization and implementation of these accountability mechanisms in an STS. Based on the proposed concept and differentiation, it is now possible to derive concrete accountability mechanisms based on data flow and information exchange. Trivially, these mechanisms need to be tailored to meet the requirements of a specific domain. We consider our approach as a step towards a unified understanding of data accountability, which can serve as a solid foundation for future research and applications.

Acknowledgments. This work is part of TUM Living Lab Connected Mobility (TUM LLCM) project and has been funded by the Bayerisches Staatsministerium für Wirtschaft und Medien, Energie und Technologie (StMWi).

References

1. Trist, E.L.: The evolution of socio-technical systems : a conceptual framework and an action research program. Ontario Quality of Working Life Centre (1981) On cover: Issues in the quality of working life : a series of occasional papers ; no. 2.
2. Bonazzi, R., Hussami, L., Pigneur, Y.: Compliance management is becoming a major issue in is design. In: Information Systems: People, Organizations, Institutions, and Technologies: ItAIS:The Italian Association for Information Systems. Physica-Verlag HD, Heidelberg (2010) 391–398
3. Feigenbaum, J., Jaggard, A.D., Wright, R.N.: Towards a formal model of accountability. In: Proceedings of the 2011 Workshop on New Security Paradigms Workshop. NSPW '11, New York, NY, USA, ACM (2011) 45–56
4. Weitzner, D.J., Abelson, H., Berners-Lee, T., Feigenbaum, J., Hendler, J., Sussman, G.J.: Information accountability. Commun. ACM **51**(6) (June 2008) 82–87
5. Banks, D.L.: The health insurance portability and accountability act: Does it live up to the promise? J. Med. Syst. **30**(1) (February 2006) 45–50
6. Weill, P., Ross, J.: IT Governance: How Top Performers Manage IT Decision Rights for Superior Results. Harvard Business School Press, Boston, MA, USA (2004)
7. Khatri, V., Brown, C.V.: Designing data governance. Commun. ACM **53**(1) (January 2010) 148–152
8. Buneman, P., Khanna, S., Tan, W.c.: Why and Where: A Characterization of Data Provenance. In: In ICDT. (2001) 316–330
9. Bechtold, S., Perrig, A.: Accountability in future internet architectures. Commun. ACM **57**(9) (September 2014) 21–23
10. Gössler, G., Le Métayer, D.: A General Trace-Based Framework of Logical Causality. In: FACS - 10th International Symposium on Formal Aspects of Component Software - 2013, Nanchang, China (2013)
11. Halpern, J.Y., Pearl, J.: Causes and explanations: A structural-model approach — part 1: Causes. CoRR **abs/1301.2275** (2013)
12. Eriksén, S.: Designing for accountability. In: NordiCHI '02: Proceedings of the second Nordic conference on Human-computer interaction, New York, NY, USA, ACM (2002) 177–186
13. Gajanayake, R., Iannella, R., Sahama, T.: Sharing with care: An information accountability perspective. IEEE Internet Computing **15**(4) (2011) 31–38
14. Gajanayake, R., Sahama, T., Iannella, R.: Principles of information accountability: An ehealth perspective. Int. J. E-Health Med. Commun. **5**(3) (July 2014) 40–57
15. Lankhorst, M.: Enterprise Architecture at Work: Modelling, Communication, and Analysis. 1st edn. Springer-Verlag New York, Inc., New York, NY, USA (2005)
16. Jonkers, H., Lankhorst, M., Buuren, R.V., Bonsangue, M., Torre, L.V.D.: Concepts for modeling enterprise architectures. International Journal of Cooperative Information Systems **13** (2004) 257–287
17. Waltl, B., Reschenhofer, T., Matthes, F.: Data governance on EA information assets: Logical reasoning for derived data. In: Workshops Proceedings of CAiSE 2015 International. (2015) 401–412