Identifying Privacy Risks in Distributed Data Services: A Model-Driven Approach

Paul Grace, Daniel Burns, Geoff Neumann, Brian Pickering, Panos Melas, Mike Surridge
IT Innovation Centre
University of Southampton, UK
(pjg, dkb, gkn, jbp, pm, ms)@it-innovation.soton.ac.uk

***Abstract*—Online services are becoming increasingly data-centric; they collect, process, analyze and anonymously disclose growing amounts of personal data. It is crucial that such systems are engineered in a privacy-aware manner in order to satisfy both the the privacy requirements of the user, and the legal privacy regulations that the system operates under. How can system developers be better supported to create privacy-aware systems and help them to understand and identify privacy risks? Model-Driven Engineering (MDE) offers a principled approach to engineer systems software. The capture of shared domain knowledge in models and corresponding tool support can increase the developers' understanding. In this paper, we argue for the application of MDE approaches to engineer privacy-aware systems. We present a general purpose privacy model and methodology that can be used to analyse and identify privacy risks in systems that comprise both access control and data pseudonymization enforcement technologies. We evaluate this method using a case-study based approach and show how the model can be applied to engineer privacy-aware systems and privacy policies that reduce the risk of unintended disclosure.**

***Index Terms*—Privacy, Cloud, Risk, Model-driven engineering**

I. Introduction

Motivation. Online services are becoming increasingly data-centric, and a growing amount of personal information about the user is collected, processed and analyzed. For the purpose of this paper, privacy is defined as the ability of a user to have control over their personal information, where users will have different viewpoints of such privacy [1], i.e. one user may care about keeping a piece of data private, whereas another user may not care if the same data is made public. However, as systems grow in complexity it becomes difficult for developers who may not be experts in the domain of privacy to ensure that such privacy properties are maintained. How can developers identify the risks of potential privacy breaches through unwanted/accidental disclosure (driven by their design and implementation choices) that contradicts the user's control of their personal information?

Contribution. In this paper we focus on using system models to: i) identify privacy risks during the development of an online service, and ii) also monitor the privacy risks during the lifetime of the service (as the users, data, and behaviour may change). We argue that such a Model-Driven Engineering (MDE) approach provides a principled method to engineer solutions in order to ensure that privacy risks are managed at different stages of system development. This is because, the capture of shared domain knowledge (in privacy models) can aid understanding across development teams, and also support automated analysis of the system to identify privacy risks. This paper proposes the following contributions:

- A *formal model of user privacy* captured as a Labelled Transition System (LTS); here states in the model represent a user's state of privacy, and labelled transitions between states represent actions on the data. This model is automatically generated from the design artifacts curated during the system design phase.
- A *data-flow driven modelling* framework. System developers specify their system in terms of a purpose-driven data-flow diagram and a set of access policies. This framework automatically generates the formal model, such that multiple analyses can be carried out upon it.
- *Automated risk analysis*. The formal privacy model is analysed to identify any privacy risks in the system's operation. The results can then be used by system designers to inform further design or system operation decisions.

We evaluate these contributions using a case-study based approach to highlight the frameworks ability to abstract a system's behaviour with regards to personal data and then machine analyze the privacy risks. We demonstrate that the privacy risks can be generated using this model-driven approach which can then inform the design of the system.

Structure. In Section II we define the data-flow modelling framework which generates the formal model of user privacy. Then in Section III the risk analysis method is presented. An evaluation of the approach is provided is Section IV. Finally, Section V examines the state of the art in privacy modelling and analysis, and we draw conclusions in Section VI.

II. Modelling Privacy Aware Systems

In this section we describe a framework to model privacy aware systems. This follows two steps. First, the developer models their system; second, a formal model of user privacy in this system is generated. Both systems under development and existing systems can be modelled in this way.

A. Step 1: modelling a privacy aware system

The developers of the system create a set of artifacts that model the behaviour and security properties of their system:

- A set of *Data-Flow diagrams* that model the flow of personal data within a system. In particular focusing on how

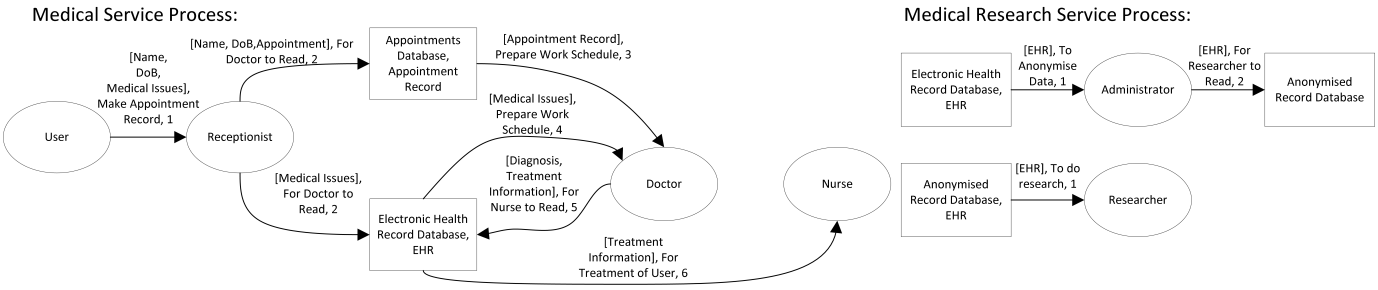

Fig. 1. Data-flow Diagrams for an example healthcare service.

data is exchanged between the actors and datastores. We utilise data-flow diagrams because they are an existing well-understood form of modelling that simply captures data behaviours.

- The *data schema* and *access control policies* associated with each datastore. That is a description of what data is stored, and which actors have access to that data. For the purpose of this paper, we assume traditional access control lists and role-based access control; however, we seek to extend the approach to consider alternative forms of access control.

We now consider a use-case to illustrate how these elements are created in practice (in this case a doctors' surgery). Two data-flow diagrams from this example are given in Fig. 1. The nodes represent either an `actor` (oval) or a `datastore` (rectangle). The datastores are labelled by two objects: the first is the identifier for the datastore, and the second are the data schemas. The actual data flow is represented by directed arrows between the ovals and rectangles, henceforth referred to as flow arrows. Each flow arrow is labelled with three objects: the set of data fields which flows between the two nodes, the purpose of the flow, and a numeric value indicating the order in which the data flow is executed. We assume datastore interfaces that support querying and display of individual fields (as opposed to coarse-grained records).

B. Step 2: automatically generating an LTS privacy model

In this section we provide a formal model of user privacy. User privacy is modelled in terms of how actor actions on personal data change the user's state of privacy. We define an `actor` to be an individual or role type which can identify the user's personal data. Depending on the service provided, each actor may or may not have the capability to identify personal data. Hence, a user's privacy changes if any of their personal data has been or can be identified by an actor. Prior models following this approach are: a Finite State Machine (FSM) [2] [3] or a Labelled Transition System (LTS) [4]. The common theme in both is that the user's privacy at any point in time is represented by a state, and that actions, executed by actors, taken on their personal data can change this state. We build upon these approaches and extend them to label both states and transitions in such a way that the model can be analysed to understand how, and why, the user's privacy

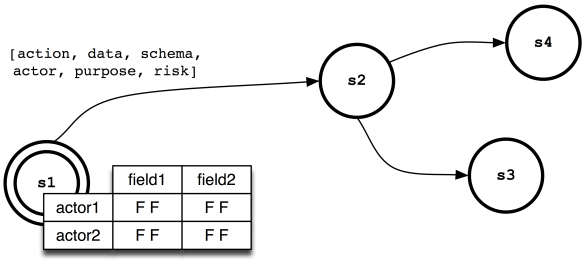

Fig. 2. A State-based Model of User Privacy

changes. This novel contribution allows us to represent not only the sharing of a user's personal information, but also the potential for a user's personal information to be shared. This is the case when personal information is stored in a datastore that can be accessed by multiple individuals.

The key elements of our model (illustrated in Fig. 2) are:

- *States*: are representations of the user's privacy. They are labelled with variables to represent two pre-dominant factors: whether a particular actor *has* identified a particular field, or whether an actor *could* identify a field. These variables, henceforth known as *state variables*, take the form of Booleans, and there are two for each actor-data field pair (has, could). The state label s1 is given the table values shown in Fig. 2; states s2, s3 and s4 have tables with different values to represent different privacy states.
- *Transitions*: represent `actions` (collect, create, read, disclose, anon, delete) on personal data performed by `actors`. They are labelled according to i) an `action`, ii) the set of `data fields`, iii) the `data schema` that the data field is a part of, iv) the `actor` performing the action. There are two optional fields: i) a `purpose` that explains the reason a particular privacy action is being taken, and ii) a `privacy risk measure` to identify risks associated with this action (whose value is calculated and annotated during risk analysis).
- *Pseudonymisation*: the disclosure of pseudonymised versions of each sensitive field is modelled using the `anon` transition. State variables (i.e. can access, has accessed) can be declared on these fields in the same way as for standard fields. For example an analyst may have access permission for the field $weight_{anon}$ but may not have

permission to access $weight$. This will mean that they may be allowed access to pseudonymised weight data for statistical purposes but should be prevented from matching any value to an individual.

Model Generation. The LTS is generated based upon the following information from the data-flow diagrams. The *actors* present in the data flow. Here, there are five actors: Receptionist, Doctor, Nurse, Administrator and Researcher. The *data fields* present in the flow (six: Name, Date of Birth, Appointment, Medical Issues, Diagnosis, Treatment Information). There are two independent services here: a medical service, and a medical research service. The example has three *datastores*: Appointments, Electronic Health Records (EHR), and an Anonymised EHR.

From this information, one can deduce that each state must be labelled with $2 * 5 * 6 = 60$ Boolean state variables. Naturally, if one wished to visualise the state system that is generated from this, each state would have to carry sixty labelling variables; this means there are 2^{60} possible privacy states. This is why the data-flow models are central to the framework; they simplify the generated model as follows:

- If data flows between a user and an actor, then this is a `collect` action, where the actor is collecting the information from the user.
- If data flows between two actors, then this is `disclose`.
- If data flows from an actor to a datastore, this is a `create` action. Where it is an anonymized data store then this is an `anon` action
- If data flows from a datastore to an actor, this is a `read` action. This is similar to the case of the `create` action; the effect on the privacy state depends upon how the datastore is accessed.
- If there are multiple flows within a service, the flows can be executed independently, provided the start node has the correct data to flow.

Using the extraction rules above combined with the access control lists, we create a state-based system. Let us begin by only considering the Medical Service process. A visual representation of the system of states using the extraction rules above is given in Fig. 3. Note that we have suppressed the state variables for this visual representation; as mentioned earlier, each node has 60 state variables.

III. Analysing Privacy Risk

Privacy risk analysis is performed on the generated model. It takes the user privacy control requirements and annotates the model with their risk; hence there is an instance for each user. The process can be executed with running users of the system, or with simulated users in the development phase. For the purpose of this paper we consider only two dimensions of risk, but seek to expand this as the work progresses.

A. Risk of unwanted disclosure

Risk assessment has two primary dimensions: the assessment of the **impact**, and the assessment of the **likelihood** that the risk event occurs. To this end, we assume that two pieces of information about the user are available:

1) Which services the user agrees to use based on its policy.
2) The user has particular sensitivities about certain fields, represented by either a sensitivity category (low, medium, high for example), or a number which takes a value between 0 and 1 indicating how sensitive the user is to disclosure of that data. We use the quantitative measure throughout this paper, and explicitly define this quantity as the *sensitivity* of $\sigma(d)$ for the field d.

This information can be obtained directly from the user through a questionnaire (if necessary). For the first point, we have that the user has explicitly agreed that actors within the chosen services can handle their personal data for particular purposes in the course of providing that service. We shall refer to these actors as *allowed actors*. An actor not associated with those services is referred to as a *non-allowed actor*.

Impact. We make the assumption that if a user has agreed to use a service, the user is insensitive to any actor using their sensitive data. This assumption is built upon the requirement that the privacy policy of the service is clearly presented to the user. Therefore, we may write the sensitivity of a data field d relative to an actor a as $\sigma(d, a)$, where $\sigma(d, a) = 0$ if the actor is allowed, and $\sigma(d, a) = \sigma(d)$ if the actor is non-allowed.

In order to assess the impact of the disclosure of a user's personal data, we shall utilise the data field sensitivities. Therefore, we need to define two elements: the sensitivity of a collection of data fields, and the *change* in the sensitivity when a transition occurs. The sensitivity of a collection of data can be computed by making the following assertion: a collection of data fields is *only as sensitive as the most sensitive data field*. In addition, we also assume that a user will be equivalently sensitive if the data field *has been* identified or the data field *could be* identified by a *non-allowed actor*. Therefore, the definition of the sensitivity of a privacy state is the *maximum sensitivity amongst the data fields that have either been identified or could be identified*.

Now, for any transition, we assess how the sensitivity changes when the transition occurs. We define the change, as the change that occurs relative to the *absolute privacy state* (where all state variables are false). For example, consider the create action for a single field d to a datastore that a non-allowed actor has access to. The sensitivity of this action is therefore $\sigma(create) = \sigma(d)$, as the absolute privacy state has no sensitivity. We shall use the maximum sensitivity change as the measure of the *impact* dimension of risk.

Let us now address **likelihood**. In this model, each transition could have an associated probability of execution, given the initial state before the transition has occurred. This requires storing the probability for each possible initial state. If we knew nothing about the system, we would have to store the information for 2^{60} possible states per transition. However, this can be simplified as follows:

- A data flow representation of the services exists and describes known behaviour within the system, simplifying the set of possible states.

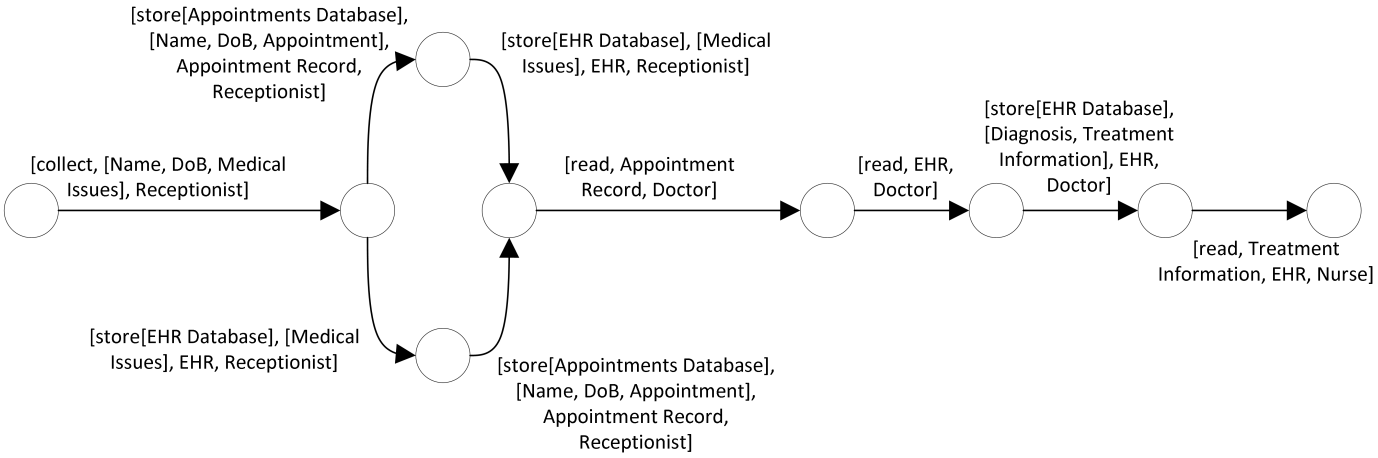

Fig. 3. The state system for the Medical Service process as an LTS.

- Specific actions allow for actors to identify personal data; namely, the `read` and `disclose` actions.
- Transitions are classified by their type.

We shall assume that the disclose action will only occur during the course of a service, and hence if a user has not agreed to use that service, the `disclose` action will not be engaged. This leaves one action: `read` that impacts the likelihood of a disclosure of a user's personal data. From the access control lists, we determine the non-allowed actors with potential access to the data. We can then attach a probability that an actor will identify a data field outside of their agreed service use. This probability can be assessed by considering the following situations:

- Accidentally access. For example, a datastore query returns a small subset of users, the actor may identify fields whilst searching for the information they need about a different user.
- If an actor maintaining the service needs to delete the data, the system may first show the data to be deleted.
- If an actor begins the execution of a service that the user did not agree to use. In this instance, one must then look at the full service and determine the implications on the user's privacy using the generated state model.

Each of these scenarios will have a probability that they will happen; the resulting probability will be the sum of the probabilities of these scenarios occurring, as they are intrinsically uncorrelated situations.

Once both the dimensions of risk are established, we can attach a risk label associated with the `read` action. For this purpose, we categorise the impact and likelihood into categories (low, medium and high), and then use a table to determine a risk level. The categorisation of the impact and likelihood, as well as the table to determine the risk level, should be specified according to the type of service.

B. Pseudonymisation Risks

We now discuss how pseudonymisation risk information is added to the LTS model for the developers chosen method of pseudonymisation. There are two key types of risk:

1) *Re-identification*: The risk that a person whose personal data is pseudonymised within a disclosed data set can be re-identified.
2) *Value*: Risk of a sensitive value being matched to an individual. Techniques such as k-anonymization [5] prevent re-identification but do not guarantee that there is not still a value risk. For example, if after k-anonymization a k-set about human physical attributes contains 10 records, 9 of which have a weight over 100kg, if an non-allowed actor knows their target is in that k-set they can be 90% certain the target has a weight over 100kg.

In this version of the model, we focus on *value* risk. A risk that a given actor (a) can access a given sensitive field (f) is said to be present in every state in the LTS where the pseudonymised version of f (f_{anon}) has been accessed by a. If a only has access rights to f_{anon} and not f, transitions will be added to the LTS starting from each of these at-risk states. For these transitions (referred to as *risk-transitions*) we calculate risk scores or declare policy associated with these transitions.

It is important to identify that the approach is to model the risk associated with a choice of pseudonymisation. For example, the above is a risk of k-anonymization that is removed when l-diversity [6] is considered; hence, we are modelling these properties not proposing a solution that is akin to l-diversity.

Calculating Risk Scores. For each transition a risk score is calculated. Consider a `read` transition on f from a state where f_{anon} has already been read. We call this state, N.

1) The anonymised fields which have already been read at N are collected together as the input field set $fields_{read}$.
2) The fields not in $fields_{read}$ are masked and the data is divided into sets, each of which contain only records which now appear to be identical.
3) An individual value risk score is calculated for f in each record. This score is the marginal probability of the value associated with an individual record (r) for f from within its set s. It is calculated as $risk(r, f) =$

$frequency(f)/size(s)$ where $frequency(f)$ is the number of occurrences of the value associated with `f` and `r` within `s` and size is the total size of that set. Note, an occurrence of a given value does not require exact equality. A user may specify a range so that $frequency(f)$ is the number of values in s which are close enough to the original value.

Using Risk Scores. The Risk score described above can only be calculated when data is present. Hence, simulated data can be used at design time, whereas the model can be applied to the running system to get a more accurate picture of risk. The risk score is used to choose pseudonymisation techniques or find out if a technique provides acceptable risk versus data utility. The resulting pseudonymised dataset with values removed can be tested for utility, by comparing statistical qualities like means and variances between the original data and the pseudonymised data. If a technique requires too much data removal and utility is shown to be likely adversely affected, the technique used would clearly be not appropriate.

IV. Evaluation

We present and evaluate two simple case studies to provide initial evidence of the benefits of the model-driven methods.

A. Identifying unwanted disclosure

Here, we considered a user in the doctors' surgery example. This user agreed to use the Medical Service, but not the Medical Research Service. We profiled the user to be sensitive about the Diagnosis field, such that the impact of that personal data being read by a non-allowed actor was High.

We used the framework and created models to generate an LTS upon which risk-analysis was performed. This first determined the actors that are non-allowed (the Administrator and Researcher), as they are not involved in the provision of the Medical Service. To highlight an unwanted disclosure, the LTS showed that the Administrator has read access to the EHR datastore after the user has used the Medical Service. Using the example risk table, the transition is labelled with a risk level of Medium for this event occurring. This risk level may be deemed unacceptable if one is designing a system with privacy in mind. The access policies were changed accordingly and the risk level was reduced to Low for this event.

In practice, this case shows there is no need to explicitly draw a formal state model. The visualisation occurs at the data-flow diagram level, and the analysis of the system can occur using the state-based model. The primary advantage of the output state model is that a developer can determine which actors can identify which data during the course of a service (in conflict with user preferences) and in turn engineer systems that assure the data subject of the transparency of any processing of their data. If such information is returned to users; identifying the risks associated with any processing enables greater understanding by the data subjects which in turn would encourage them to take responsibility themselves for their own data Hence, there is the potential for the information output from the analysis to form part of the privacy policy explained to users.

TABLE I
Risk values for 2-anonymisation data records

Age	Height (cm)	Weight (kg)	Height risk	Age risk	Age Height risk
30-40	180-200	100	2/4	2/2	2/2
30-40	180-200	102	2/4	2/2	2/2
20-30	180-200	110	2/4	3/4	2/2
20-30	180-200	111	2/4	3/4	2/2
20-30	160-180	80	1/2	1/4	1/2
20-30	160-180	110	1/2	3/4	1/2
Violations:			0	2	4

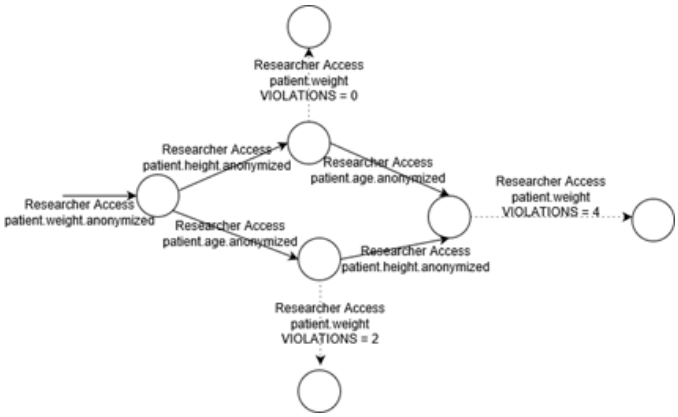

Fig. 4. Pseudoanonymisation risk analysis output

B. Identifying pseudomysation risk

For this case, we prepared the health record datastore records to undergo 2-anonymisation. A researcher then has access to this data but does not have access to the original data. The policy violation that we wish to avoid is the researcher being able to predict an individual's weight to within 5kg with at least 90% confidence. Age and height are quasi identifiers. In these situations we can say that the risk of value re-identification is over 90%. Table I provides six sample records input to the model analysis process and shows how, as more identifying fields become available to the researcher, the number of violations of this policy increases.

Our system is represented in the generated LTS (Fig. 4). Dotted lines indicate potential policy violations. A system administrator has the option of loading the six records given as examples above into the LTS. They would then see the violation scores 0, 2 and 4 as shown in this figure. This may cause them to consider increasing their k value or reconsider their pseudonymisation entirely. Alternatively, at the design phase, a system designer could declare that a number of violations above 50% is unacceptable. The system would now throw an error if the above data was used, forcing the administrator to choose another form of pseudononymisation.

V. Related Work

Privacy Modelling. Both Fischer [3] and Kosa [2] define formal models of privacy in terms of state machine representations. Their purpose to demonstrate that a system complies with privacy regulations and requirements. Such models offer strong building blocks that our formal privacy model builds upon; in particular moving from hand-crafted specifications to auto-generated models that underpin the privacy engineering process and privacy risk analysis. MAPaS [7], is a model-based framework for the specification and analysis of privacy-aware systems. Centred upon a UML Profile, purpose-based access control systems are modelled and the framework allows queries to be executed to identify errors in the design.

Privacy Risk Analysis. LINDDUN [8] is a framework for performing privacy threat analysis on the design of a system in order to select privacy enforcing mechanisms that mitigate the threats. This combines a data flow model of the system with a privacy threat catalogue to provide a design-time methodology to assess privacy risks. We similarly employ a data-flow oriented methodology but explore the extent risk can be analysed automatically via the generation of an underpinning formal model. Further, we consider the use of MDE methods beyond the design phase (and in particular analysis of running systems with real users). The increasing prevalence of data anonymisation adds different types of privacy risk [9]. There are a number of tools available to anonymise data, which also provide some risk analysis feedback. The ARX Tool [10] provides methods for analyzing re-identification risks following the prosecutor, journalist and marketer attacker models on a number of anonymisation algorithms. The Cornell Anonymization Toolkit (CAT) [11] performs Risk Analysis in terms evaluating the disclosure of risks of each record in anonymised data based on user specified assumptions about the adversarys background knowledge. These tools offer important insights to identify privacy risks; and in our approach we seek to integrate similar capabilities into our methodology.

Privacy Policy Analysis. A system's behaviour should be matched against it's own privacy policy. [12] models a system's behaviour in terms of a Business Processing Model Notation (BPMN) diagram and then the goal is to check whether this is compliant with the system's P3P privacy policy. [13] integrate links to the privacy policy in the system's workflow (e.g. the BPEL specification), these are then checked by an analysis tool at design time to determine if the workflow agrees with the policy. [14] provide a similar method; rather than having a designer merge the workflow and policy, the approach converts both models (a BPEL specification and P3P policy) into a graph representation before formally analyzing the correctness of the graph. However, all of these solutions only check if a system behaves according to its stated privacy policy (our LTS can be similarly analysed); there has been limited research into the evaluation of a system in terms of privacy risk. The method presented in this paper seeks to better unify the design of a system with its privacy policy while also allowing the engineers the ability to identify and mitigate against potential privacy risks.

VI. Conclusions and Future Work

In this paper, we have proposed the consideration of model-driven methods to support the design and engineering of privacy-aware systems by developers who may not be privacy experts. Specifically, we have presented a framework to create design models that can be used to generate a formal model of user privacy that can then be analysed to identify risks including unwanted disclosure of sensitive personal data, and sensitive information inferred from pseudonymysed data.

Our preliminary evaluation highlights the potential of the approach; and therefore the areas of future work will seek to concretise the methods further via the development of better tool support that will then be utilised by real-world system developers. Our research methodology (using qualitative methods involving system developers) will then seek to answer the questions: to what extent does the tool and method reduce the effort of a developer in creating a privacy-aware system; and how valuable is the information highlighted by the analysis.

Acknowledgment

This work was supported by the European Commission under the Horizon 2020 Programme (H2020), as part of the OPERANDO project (Grant Agreements no. 653704).

References

[1] P. Kumaraguru and L. F. Cranor, "Privacy indexes: a survey of westin's studies," 2005.

[2] T. A. Kosa, "Towards measuring privacy," Ph.D. dissertation, University of Ontario Institute of Technology, 4 2015.

[3] S. Fischer-Hbner and A. Ott, "From a formal privacy model to its implementation," in *Proceedings of the 21st National Information Systems*, 1998.

[4] P. Grace and M. Surridge, "Towards a model of user-centered privacy preservation," in *Proceedings of the 12th International Conference on Availability, Reliability and Security*, ser. ARES '17. New York, NY, USA: ACM, 2017, pp. 91:1–91:8.

[5] L. Sweeney, "K-anonymity: A model for protecting privacy," *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, vol. 10, no. 5, pp. 557–570, Oct. 2002.

[6] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam, "L-diversity: privacy beyond k-anonymity," in *22nd International Conference on Data Engineering (ICDE'06)*, April 2006, pp. 24–24.

[7] P. Colombo and E. Ferrari, "Towards a modeling and analysis framework for privacy-aware systems," in *2012 International Conference on Privacy, Security, Risk and Trust and 2012 International Confernece on Social Computing*, Sept 2012, pp. 81–90.

[8] M. Deng, K. Wuyts, R. Scandariato, B. Preneel, and W. Joosen, "A privacy threat analysis framework: supporting the elicitation and fulfillment of privacy requirements," *Requirements Engineering*, vol. 16, no. 1, pp. 3–32, 03 2011, copyright - Springer-Verlag London Limited 2011; Document feature - ; Last updated - 2014-08-30.

[9] S. Ying and T. Grandison, "Big data privacy risk: Connecting many large data sets," in *2016 IEEE 2nd International Conference on Collaboration and Internet Computing (CIC)*, 2016.

[10] F. Prasser and F. Kohlmayer, *Putting Statistical Disclosure Control into Practice: The ARX Data Anonymization Tool*. Cham: Springer International Publishing, 2015, pp. 111–148.

[11] X. Xiao, G. Wang, and J. Gehrke, "Interactive anonymization of sensitive data," in *Proceedings of the 2009 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '09. New York, NY, USA: ACM, 2009, pp. 1051–1054.

[12] M. Chinosi, A. Trombetta *et al.*, "Integrating privacy policies into business processes," *Journal of Research and Practice in Information Technology*, vol. 41, no. 2, p. 155, 2009.

[13] S. Short and S. P. Kaluvuri, "A data-centric approach for privacy-aware business process enablement," in *International IFIP Working Conference on Enterprise Interoperability*. Springer, 2011, pp. 191–203.

[14] Y. H. Li, H.-Y. Paik, and B. Benatallah, "Formal consistency verification between bpel process and privacy policy," in *Proceedings of the 2006 International Conference on Privacy, Security and Trust*. ACM, 2006, p. 26.