Eddy *a privacy specification language*

Found at: https://goo.gl/EYmYXQ

Check the URL above for access to the eddy tool demonstration and tutorials, which provides a web-based interface to the underlying Java tool.

Eddy is a privacy requirements specification language that privacy analysts can use to express requirements over acts to collect, use, transfer and retain personal and technical information. The language uses a simple SQL-like syntax to express whether an action is permitted or prohibited, and to restrict those statements to particular data subjects and purposes. The Eddy specifications are compiled into Description Logic to automatically detect conflicting requirements and to trace data flows within and across specifications. Each specification can describe an organization's data practices, or the data practices of specific components in a software architecture.

Recommended Reading Materials

[1] Eddy, A Formal Language for Specifying and Analyzing Data Flow Specifications for Conflicting Privacy Requirements
Travis D. Breaux, Hanan Hibshi, Ashwini Rao. *Requirements Engineering Journal,* 19(3): 281-307, 2014.

Download the paper at: http://goo.gl/WHNzG0

This an extended journal version of our conference paper that was nominated for best paper and presented at IEEE RE'13. It describes the original formulation of the eddy language, and provides a detailed description of the privacy and data supply-chain concept, the case study that was used to express the design for the language, and the research method which was expounded upon in [2].

For more on the Description Logic framework that underlies the OWL ontology expressed by the eddy language, see section *3.1. Introduction to Description Logic*.

For more on the formulation of a privacy requirements specification, including the definition of entities in the eddy language (Datum, Purpose, and Actor), see section *3.2. Privacy Requirements Specifications*. The mechanism by which requirements conflicts can be detected is explained in section *3.2.1. Detecting Requirements Conflicts*. This section also explains the classification of data flows, as *underflow, overflow, exact flow, and no flow*, with accompanying formulations in description logic; we can trace permitted data collections (source actions) to permitted data uses and data transfers (target actions). Underflow occurs when a data source is subsumed by a target, overflow occurs when a data target is subsumed by

the source, exact flow occurs when the data source and target are in alignment, and no flow occurs otherwise.

The grounded analysis coding method used to extra data requirements for expression using eddy is described in section *4. Coding Method to Extract Data Requirements*. Be sure to take a quick look at Figure 6 (see right) and Figures 7 through 9 (not pictured) for some helpful pictorial descriptions of the process.

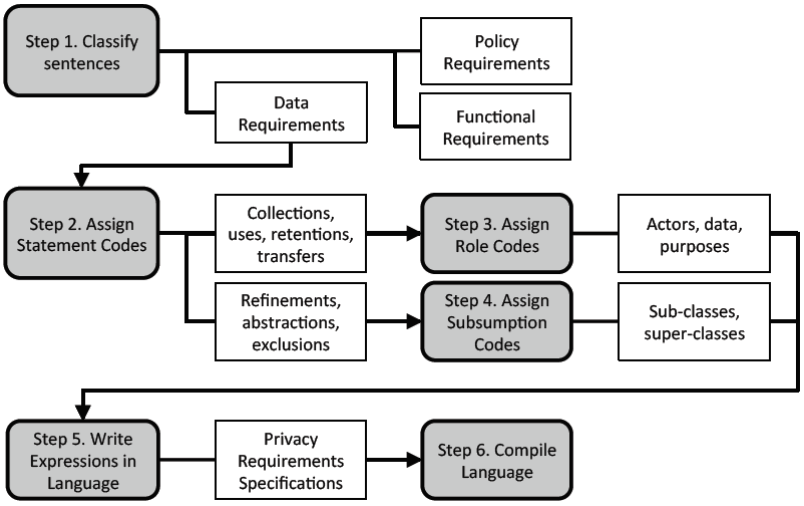

Figure 6, taken from ***Eddy, A Formal Language for Specifying and Analyzing Data Flow Specifications for Conflicting Privacy Requirements***

Section *5. Case Study Results* presents the results of the paper's case study, but pay close attention to section *5.2. Conflicts Identified Using the Eddy Language*, as it highlights one of the key contributions of the eddy language; the ability to automatically reason over the complex policies present in the case study, and identify where conflicting policies emerge. The tool reasons about and highlights the requirements that are in conflict with one another automatically.

For more information on the potential scalability of the tool for large policies, as well as a discussion of the performance of the underlying DL inference engine, see section *6. Simulation Results.* Here, artificial datasets were used to determine how quickly the eddy language toolset can reason over increasingly large policies/ontologies. We show that the tool scales well to policies that are larger than those expressed in our case study, requiring less than a minute to reason over most policies.

[2] Detecting Repurposing and Over-collection in Multi-party Privacy Requirements Specifications
Travis D. Breaux, Daniel Smullen, Hanan Hibshi. To Appear: *23rd IEEE International Requirements Engineering Conference*, Ottawa, Canada, 2015.

Download the paper at: http://goo.gl/Cqr4An

This conference-length paper is a new extension of the original work on eddy, in which a new case study is used to explore the concepts of repurposing and information over-collection. In this study, the eddy tool is applied to multi-party privacy requirements specifications; the Waze application, in conjunction with its mobile analytics and advertising partners, scalable storages services, and authentication service providers is analyzed.

The *Contributions* section on the first page highlights the new techniques used in the paper, but most importantly it describes the three critical privacy principles which are explored in the paper: the **purpose specification principle**, which requires that purposes for which data is collected should be explicitly stated, the **collection limitation principle**, which requires that collection of personal and/or protected data should be limited as much as possible to be useful, and the **use limitation principle**, which requires

that expressed uses should be limited to the purposes for which data was original collected, and nothing further – with exceptions for explicit consent and legal compliance.

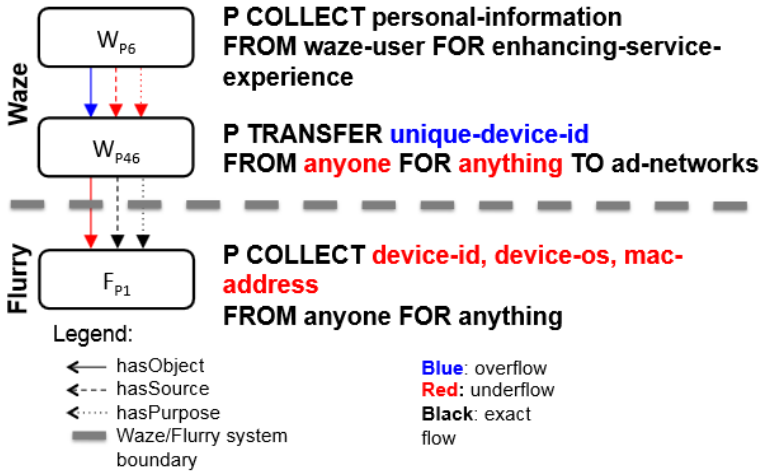

*Figure 3, taken from **Detecting Repurposing and Over-collection in Multi-party Privacy Requirements Specifications***

For more on the formulation of multi-party data flows, be sure to read Definitions 1 – 3 in section *1. B. New Extensions to Privacy Specifications*. Here, we describe the new infrastructure required to synchronize the definition of terms between multiple parties using a shared lexicon, also referred to as a dictionary. Service maps describe the relationships between two agents, and their data transfer requirements. Multi-party traces map transfer requirements for data to collection requirements across multiple parties' specifications. Figure 3 (see left) is helpful to show this pictorially.

For more information on the conflicts discovered within the Waze case study, have a look at section *5. A. Potential Conflicts within Each Policy*. Here, the results of eddy's automated analysis are shown, and there is discussion (with accompanying eddy syntax) on how to approach resolution of ambiguity related, electable permission related, and direct conflicts.

Section 5. C. Verifying the Three Privacy Principles explores the presence of these privacy preserving actions within the policies taken from Waze. Discussion of missing purposes is seen in paragraph 2, analysis of repurposing is seen in paragraph 3, and discussion of over-collection is seen in paragraph 4. The discussion in paragraph 5 describes two privacy specification design patterns that can be used to bypass the limitation principles – this is done by writing policies that violate the general spirit of the principles, while still complying with the policy. The first is called **purpose hoisting**, which describes permitted actions with restricted purposes. These actions comply with purpose specifications, and under the limitation principle constrain any target actions. The author then describes the same permitted action with a broader purpose, subsuming the original purposes, and hosting the information from a restricted purpose to a more general purpose. The second is called **unrestricted cross-flows**, which describes the source and target action purposes in general terms to offer greater design flexibility, but at the cost of privacy. This flexibility benefits companies who want to evolve their business models without modifying policies, but introduces privacy risks.

Finally, a novel performance analysis was performed and can be seen in section *5. D. Tool Support for Scaling Multi-Party Compositions*, which sought to determine whether the even larger policies characteristic of multi-party compositions would be workable using the tool. Our results showed that even with hundreds of requirements/rule statements, policy sizes showed quasilinear scalability with respect to time. We also showed that for increasing numbers of data expressed in the eddy tool, similar quasilinear scalability was evident.

Eddy Quick Reference Guide	
Syntax	**Example**
SPEC HEADER P (purpose): What are we doing with this data? D (datum): Data entities. A (actor): Who is involved in using the system? SPEC POLICY R (right): What is allowed? P (prohibition): What is not allowed? Actions: COLLECT, USE Other keywords: FROM, FOR	R USE bloodwork FROM phlebotomist FOR marketing • "You're allowed to use bloodwork from the phlebotomist for marketing purposes." R COLLECT bloodwork FROM laboratory FOR treatment • "You're allowed to collect bloodwork from the laboratory for treatment purposes." Relationships expressed in the header: P treatment > diagnosis, prescription, bloodtests "The purpose of treatment also means diagnoses, prescription, and bloodtests."