SPECIAL

Scalable Policy-awarE Linked Data arChitecture for prIvacy, trAnsparency and compLiance

Deliverable D2.3

Transparency Framework V1

Document version: V1.0

SPECIAL DELIVERABLE

Name, title and organisation of the scientific representative of the project's coordinator:

Ms Jessica Michel t: +33 4 92 38 50 89 f: +33 4 92 38 78 22 e: jessica.michel@ercim.eu

GEIE ERCIM, 2004, route des Lucioles, Sophia Antipolis, 06410 Biot, France

Project website address: http://www.specialprivacy.eu/

Project	
Grant Agreement number	731601
Project acronym:	SPECIAL
Project title:	Scalable Policy-awarE Linked Data arChitecture for prIvacy, trAnsparency and compLiance
Funding Scheme:	Research & Innovation Action (RIA)
Date of latest version of DoW against which the assessment will be made:	17/10/2016
Document	
Period covered:	M1-M23
Deliverable number:	D2.3
Deliverable title	Transparency Framework V1
Contractual Date of Delivery:	28-02-2018
Actual Date of Delivery:	28-02-2018
Editor (s):	Sabrina Kirrane (WU)
Author (s):	Sabrina Kirrane (WU), Uroš Milošević (TF), Javier D. Fernández (WU), Axel Polleres (WU)
Reviewer (s):	Rigo Wenning (ERCIM), Piero Bonatti (CeRICT), Wouter Dullaert (TF)
Participant(s):	WU, TF, ERCIM, CeRICT
Work package no.:	2
Work package title:	Policy and Transparency Framework
Work package leader:	WU
Distribution:	PU
Version/Revision:	1.0
Draft/Final:	Final
Total number of pages (including cover):	40

Disclaimer

This document contains description of the SPECIAL project work and findings.

The authors of this document have taken any available measure in order for its content to be accurate, consistent and lawful. However, neither the project consortium as a whole nor the individual partners that implicitly or explicitly participated in the creation and publication of this document hold any responsibility for actions that might occur as a result of using its content.

This publication has been produced with the assistance of the European Union. The content of this publication is the sole responsibility of the SPECIAL consortium and can in no way be taken to reflect the views of the European Union.

The European Union is established in accordance with the Treaty on European Union (Maastricht). There are currently 28 Member States of the Union. It is based on the European Communities and the Member States cooperation in the fields of Common Foreign and Security Policy and Justice and Home Affairs. The five main institutions of the European Union are the European Parliament, the Council of Ministers, the European Commission, the Court of Justice and the Court of Auditors (http://europa.eu/).

SPECIAL has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731601.

Contents

List of Figures

1 Summary

The objective of the SPECIAL transparency and compliance framework is to help data processors and controllers to understand how the components and know-how developed in SPECIAL can be leveraged within a typical Enterprise setting. Core components of the SPECIAL transparency and compliance framework include: (i) the schema and vocabularies that can be used to express usage policies and data processing and sharing events; (ii) and the corresponding usage control middleware that is needed to interact with such data; and (iii) periphery components required in order to hook into Enterprise Line of Business and Business Intelligence / Data Science systems.

This aim of this deliverable is to define the scope of the SPECIAL transparency and compliance framework, which is used to guide the implementation of the SPECIAL platform and components, and also serves as a reference point for the open research challenges that we address in SPECIAL and how they relate to one another.

This deliverable builds upon the SPECIAL policy language which is described in *D2.1: Policy Language V1*. While, related information on the compliance checking, distributed ledger technology, and big data processing can be found *Deliverable D2.4 Transparency and Compliance Algorithms V1*.

In *Chapter* 1 we provide a high level overview of the SPECIAL landscape. After setting the scene in terms of the data sources, middleware and applications, we discuss the key role of Personally Identifiable Information (PII) in terms of Data Governance and identify open challenges that we aim to address in SPECIAL in *Chapter* 2. Finally, in *Chapter* 3, we provide our initial proposal for how the Resource Description Framework can be used to represent data processing and sharing events, by describing the key terms and their relationship both to one another and to the terms specified in SPECIAL usage policies.

Considering the iterative nature of the SPECIAL project, it is worth noting that the framework, personal data inventory and the linked provenance/event information described in this deliverable are expected to evolve throughout the course of the project. Consequently, such considerations will be included in future versions of this deliverable. Later versions of this deliverable will also include additional information on robustness in terms of security, performance and scalability.

Chapter 1

Policy, Transparency and Compliance Framework

In *D1.3: Policy, transparency and compliance guidelines V1 (Chapter 6)* we identified several considerations and open questions with respect to the intersection between existing company systems and SPECIAL components. In this section, we frame the SPECIAL policy, transparency and compliance components developed in SPECIAL within the wider scope of a general Enterprise setting.

Enterprises rely on operational systems, commonly known as Line Of Business (LOB) applications, to perform day-to-day activities efficiently. For example, interactions with clients are recorded in Customer Relationship Management (CRM) applications, employee information is maintained in Human Resources (HR) applications and project documentation is stored in a Document Management Systems (DMS). When it comes to strategic decision making the data from LOB applications are ususally integrated and stored in a data warehouse that can be used for Business Intelligence (BI) / Data Science (DS) across the organisation.

1 SPECIAL Landscape

The purpose of the proposed policy, transparency and compliance framework depicted in *Figure* 1.1, and discussed below, is threefold: (i) to better understand the intersection between SPECIAL and existing company systems; (ii) to define the scope of the SPECIAL transparency and compliance work; and (iii) to serve as a framework that can be used to compare and contrast alternative solutions.

Components that are coloured in green are assumed to exist already, while components in blue will be developed by SPECIAL and/or the know how to develop said components will be provided by SPECIAL. In addition, the RDF symbol is used to denote RDF data, HDFS and Spark are used to highlight big data and big data processing respectively, and a simple reasoning symbol is used to denote components could potentially require some form of reasoning capabilities.

1.1 Data sources

The framework contains four different data sources. Two of which we assume already exist as they are necessary to support business operations (i.e. Line of Business Data), and strategic

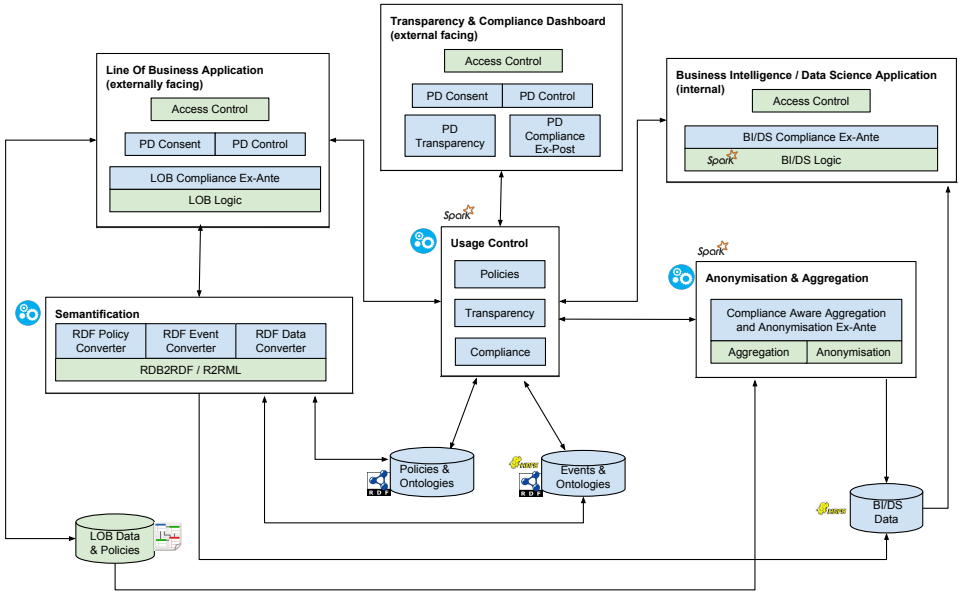

Figure 1.1: SPECIAL Landscape

decision making (i.e. Business Intelligence / Data Science Data). In addition, we propose two additional data sources one which is used to store the consent, regulatory and business *policies* and another to store the data processing or sharing *events*. Both policy and the event data in SPECIAL will be represented as RDF. In our framework we have chosen to logically represent the data in two separate data stores as it is assumed that we will need to deal with a high volume of event data, which is most likely not the case for the policy data. It's worth noting that policy information will also need to be attached to the data stored in the BI/DS Data repository.

For additional information on the policy language the reader is referred to *D2.1: Policy Language V1*. While, details the LOB and BI/DS Data can be found in *Chapter* 2 *Personal Data Inventory*, which describes the SPECIAL personal data inventory strategy. Finally, information on the event log can be found in *Chapter* 3 *Linked Provenance/Event Information* of this deliverable.

1.2 Middleware

Beside the *Usage Control* middleware which is a core component of the policy, transparency and compliance framework, we propose two additional middleware components, namely *Semantification*, and *Anonymisation & Aggregation*.

1.2.1 Usage control

The usage control middleware is responsible for managing access to policies, event data and the respective ontologies. In *D2.4: Transparency and Compliance Algorithms V1 (Chapter 2)* we explore a number of *distributed architectures* that could potentially be used to store the event data. In addition in *D2.4: Transparency and Compliance Algorithms V1 (Chapter 3)* we

introduce the SANSA Stack and discuss how it can be used to both query event data and to verify compliance of data processing an sharing events, at scale.

Irrespective of how the event data is stored (e.g. in a local, global or distributed ledger), in order to ensure non-repudiation by any of the involved peers (i.e. those owning, disclosing, and acquiring data), it must be possible to ensure that all recorded transactions have actually taken place and the autogenerated provenance/event data is tamper proof. As such we also examine the guarantees offered by *fair exchange* protocols in terms of non-repudiation of data sharing events in *D2.4: Transparency and Compliance Algorithms V1 (Chapter 4)*.

It is worth noting, that an exploration of suitable compression and encryption mechanisms will be deferred to *D2.7 Transparency Framework V1* and *Transparency and Compliance Algorithms V2*.

1.2.2 Semantification

Considering the variety of different LOB and BI/DS systems and database schemas, there is a need for RDF Converters that are capable of translating policies, events and possibly also LOB data into RDF. Well know approaches include Ontology-based Data Access (OBDA) in general and RDB2RDF and R2RML in particular. Updates over OBDA or even updates to Linked Data under ontological entailment are under-researched research topics within the Semantic Web Community. One of the main challenges where is understanding what is personal data, what policies are attached to that data and how are data processing and sharing events currently recorded, so that such information can be mapped to RDF. Here we are working on techniques that can be used for data discovery and cataloging. Additional details on the semantification of LOB and BI/DS Data can be found in *Chapter* **??** *Personal Data Inventory*.

1.2.3 Anonymisation and aggregation

Assuming that some BI/DS systems may depend on anonymisation and aggregation techniques it would be useful to be able to cater for policy-aware data anonymisation and aggregriation. The research in this area includes using machine learning techniques to verify existing anonymisation techniques, such as k-anonymity, l-diversity, t-closeness, to name but a few. This will improve the capability to avoid de-anonymisation and the ability to single out yet unknown persons for further discrimination. SPECIAL also aims to investigate how existing policies can be used to inform the anonymisation and aggregation algorithms.

Additional details on the validation of existing anonymisation techniques and policy aware anonymisation and aggregation mechanisms will be deferred to *D2.7 Transparency Framework V1* and *Transparency and Compliance Algorithms V2*.

1.3 Applications

In the proposed framework we have identified three different types of applications, namely, transparency and compliance dashboards, operational LOB applications, and strategic BI/DS applications.

1.3.1 Transparency and compliance dashboard

The transparency and compliance dashboard should be developed in such a way that it tackles the users' cognitive limitations. Key functions could include: obtaining consent in a non

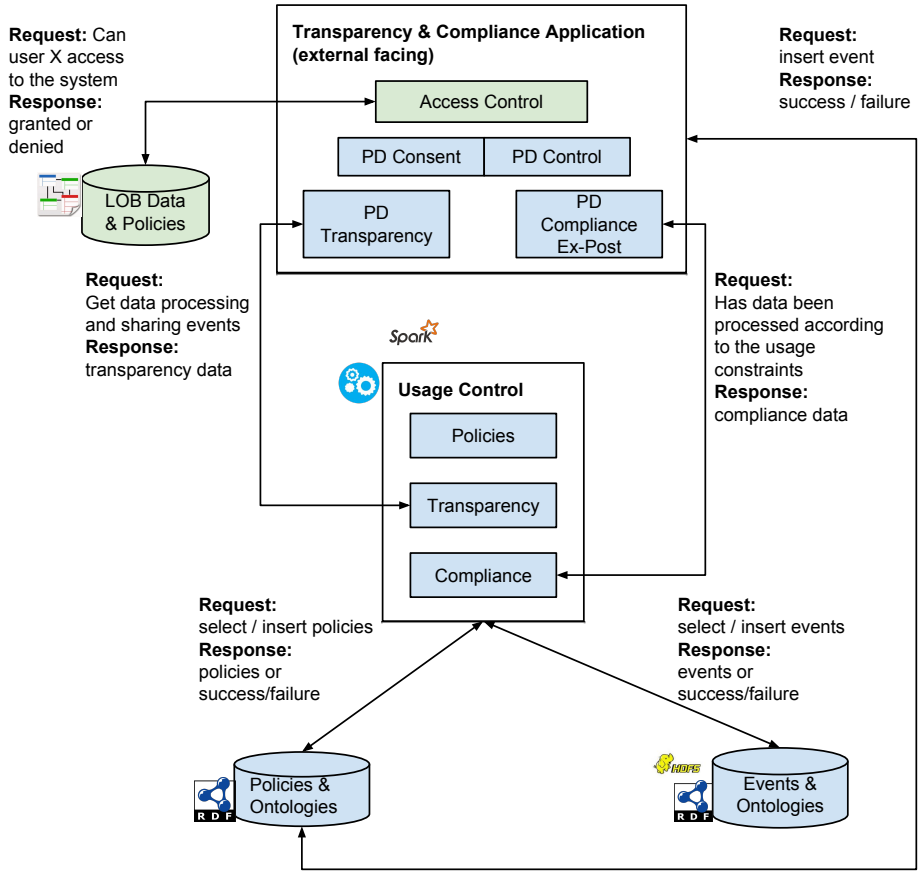

Figure 1.2: Transparency and compliance application

intrusive manner, presenting data processing and sharing events in a easily digestible manner, and enabling the user to manage existing consent for processing and sharing. *D6.3: Plan for community group and standardisation contribution* points to mashups as one potential means to support the integration of data coming from several different sources (most likely under the control of different controllers/processors). Although authentication and authorisation would highly depend on the existing company infrastructure, Web Identity and Discovery (WebID), which is a mechanism used to uniquely identify and authenticate a person, company, organisation or other entity, by means of a URI, could potentially be used for authentication across different enterprises. Highlevel details of the interaction between the Transparency & Compliance Application, the Usage Control middleware and the Policies and Events data stores are presented in *Figure* 1.2.

1.3.2 Line of Business applications

Clearly there is a tight coupling between SPECIAL and existing Line of Business applications in terms of access control, consent and compliance checking. As mentioned earlier, authentication and authorisation would highly depend on the existing company infrastructure. Additionally,

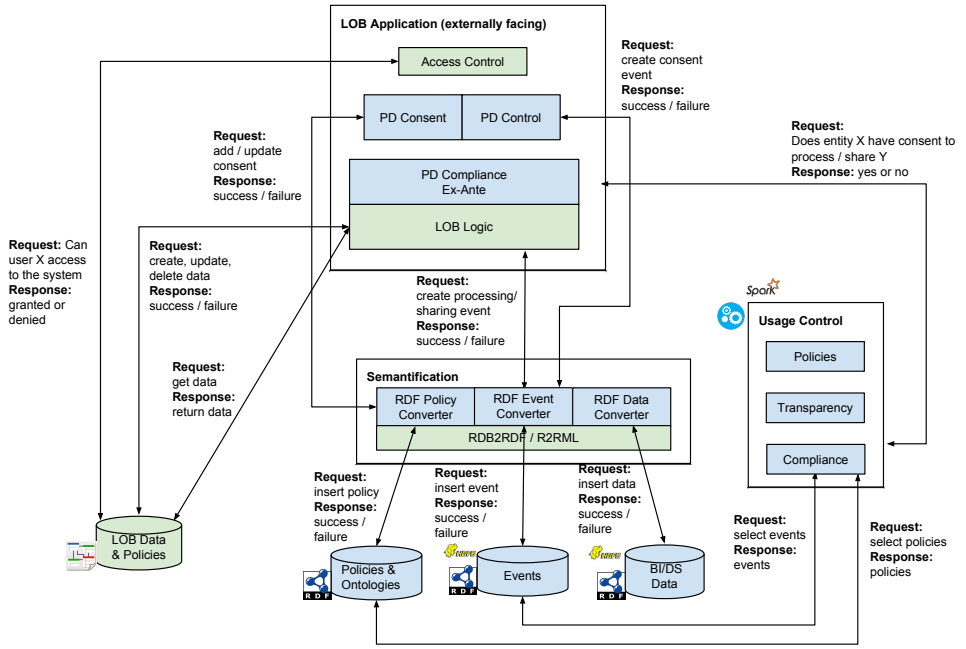

Figure 1.3: LOB application

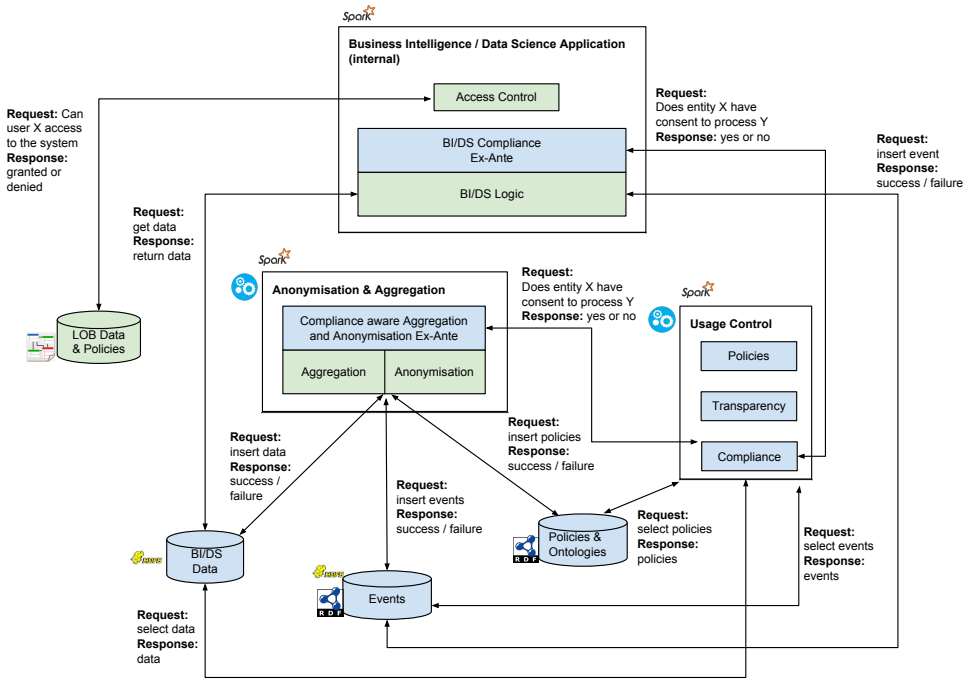

Figure 1.4: BI/DS application

the data that will form part of the consent request and subsequently the usage policy needs to be based on the type of personal data required by the company in terms of product or service provision, and contextual information relating to the purpose, processing and sharing. Similarly, companies need to ensure that personal data processing and sharing within the orgaisation and by its Information Technology (IT) systems complies with relevant usage policies. Here there is a need to investigate and come up with a strategy for hooking into existing LOB applications. Additional details can be found in *Chapter* 2 *Personal Data Inventory*. Highlevel details of the interaction between the Line of Business Application, the Usage Control and Semantification middleware, and the Policies and Events data stores are presented in *Figure* 1.3.

1.3.3 Business Intelligence / Data Science applications

As per the LOB applications there is a tight coupling between SPECIAL and existing Business Intelligence / Data Science applications, however here the focus is mainly on access control, and compliance checking. As before authentication and authorisation would highly depend on the existing company infrastructure. Additionally, companies need to ensure that personal business intelligence and data science within the orgaisation complies with relevant usage policies. Here again there is a need to investigate and come up with a strategy for hooking into existing BI/DS applications. In this deliverable, we first explore how usage control, transparency and compliance checking can be added to existing LOB applications. In *D2.7 Transparency Framework V1* and *Transparency and Compliance Algorithms V2* we will further extend on this work by proposing mechanisms to add usage control, transparency and compliance checking to BI/DS applications. Highlevel details of the interaction between the Business Intelligence / Data Science Application, the Usage Control and Anonymisation & Aggregriation middleware, and the Policies and Events data stores are presented in *Figure* 1.4.

Chapter 2

Personal Data Inventory

Semantification of data is only possible under the assumption that the controller is aware of its existence, in all its forms, as well as its precise location. One way to ensure this is by cultivating the organisational behavior necessary to successfully manage data as an asset within the company itself. However, formal Data Governance implies establishing stakeholder agreement on data definitions, developing policies and procedures, encouraging data stewardship practices at multiple levels within the organisation, and continuous and active engagement in organisational change management processes. The time and effort required to accomplish that goal, along with the accompanying cost, are the prime reasons why true Data Governance remains out of reach for many enterprises.

A less disruptive approach is to try and document the data as it is, without making any assumptions about the existing data management culture or governance processes. Instead, the organisation attempts to build a metadata repository, that is, a catalog of the company's digital assets. It does not impose any formal processes on the existing culture, and allows the digital ecosystem to change and grow. Enterprise Metadata Management[1], much like Master Data Management (MDM)[2], however, is hampered by the constant need for manual inspection and alignment of disparate data sources. In fast changing/growing environments, the inability to maintain the required pace translates into inconsistent knowledge, which consequently has a negative impact on decision making and, ultimately, results in loss of trust in the solution.

The chief downside of most popular approaches to enterprise data management is that they focus on what the enterprise considers to be *key business entities* and associated information, stored primarily in structured databases. Such approaches tend to overlook the effect of unstructured data and, more importantly, that of the information assets organisations collect, process and store during regular business activities, but generally fail to use for other purposes, also known as *dark data*[3].

1 Data Discovery

SPECIAL aims to go beyond identifying essential business entities and personally identifiable information (PII) in structured data, and investigate alternative approaches to building digital enterprise inventories. Traditional Discovery / Data Loss Prevention tools rely on manual effort,

[1] https://www.gartner.com/it-glossary/enterprise-metadata-management-emm
[2] https://www.gartner.com/it-glossary/master-data-management-mdm
[3] https://www.gartner.com/it-glossary/dark-data

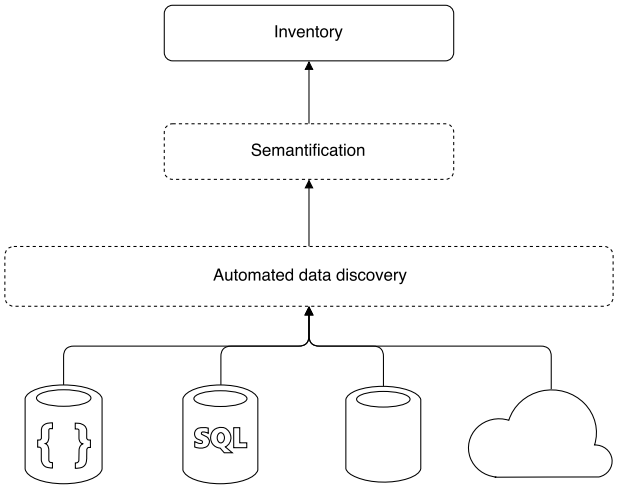

Figure 2.1: Assisted Data Discovery and Cataloging

rules and regular expressions, which lack both context and semantics. In Big Data ecosystems, such methods are often ineffective due to the sheer amount of data and the ever-present ambiguity.

1.1 Assisted data discovery

Privacy-preserving Data Mining usually assumes that the location of the private information is known. Private Information Discovery, on the other hand, attempts to identify the location and type of the private information [5]. Naming conventions aside, it is worth noting that the results obtained via the discovery process, can serve as input/constraints in the data mining process.

Similarly, a semi-automated discovery workflow relying on machine learning methods can greatly reduce the search space for identity mining by identifying pools of PII across distributed and heterogeneous data sources. Additionally, AI algorithms can help discover sensitive data attributes belonging to a specific data subject across multiple enterprise systems, and even help correlate them with attributes which were previously unknown or considered irrelevant [6].

1.2 Identity graphs

As PII is more often than not scattered across several systems, it is also not unusual to find the same piece of information in multiple data sources. Being able to identify and interlink all known sensitive data belonging to a subject, all occurrences of the same piece of information, and even the data that may not as yet have been accounted for, means being able to recreate the subject's virtual *identity graph*.

The graph must capture at least the essential knowledge on (1) the data subject, (2) the attribute, (3) the value, and (4) the location. RDF provides the flexibility to represent such information in more than one way, but also the means to easily extend it further. A straightforward solution could directly reference the attribute in question (such as `:name`, Figure 2.3). But, such a model would not allow for reusability of standard vocabularies due to accompanying cardinality and range restrictions of certain properties.

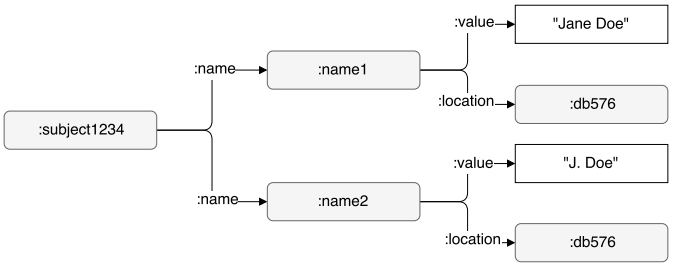

Figure 2.2: Simple identity graph representation

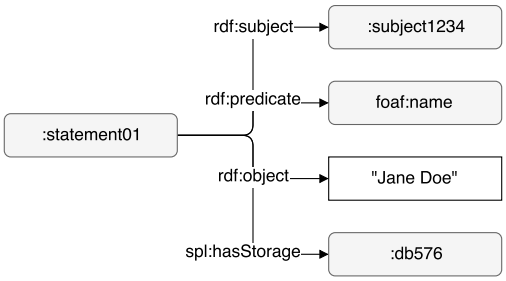

Figure 2.3: Identity graph via reification

An alternative way to encode identity provenance information is reification. An example, using the standard RDF reification vocabulary[4], is given in Figure 2.2. Such a construct, however, does not reflect the domain model, as the statement itself assumes the role of the subject. Reification via singleton properties [9], on the other hand, would keep the data subject at the heart of the model and provide a more intuitive representation of a person's identity (Figure 2.4). SPECIAL will also explore other possibilities.

A complete graph can give an overview of what truly constitutes PII, as well as help identify inconsistencies, remove duplicates and minimise risks pertaining to a potential data breach. It can also ensure that both the controller and the subject always have a clear picture of *what belongs to whom*, but also *what is correlated to what*, so individuals can exercise even more control over their own data. In the context of SPECIAL, this would translate into more detailed data lineage and enable fast and deep insights into data provenance and risk analysis.

The identity graph can be viewed as a map of the data within an enterprise; it describes the situation as it is. This can help inform data governance decisions and track progress, but, as such, these identity graphs do not change the underlying data.

1.3 Ontology Based Data Access

ETL approaches to collecting relevant data would inevitably introduce another layer of complexity, as well as additional data integrity and security risks. The derived identity graph mappings, however, could be used to create a virtualisation layer without replicating the original data, and provide limited access to authorised individuals on request.

[4]https://www.w3.org/TR/rdf-primer/#reification

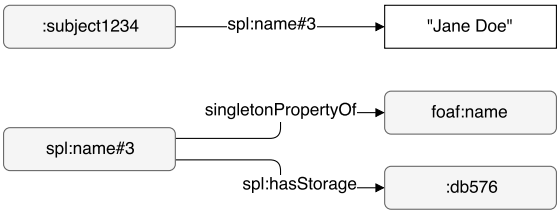

Figure 2.4: Identity graph via singleton property reification

R2RML[5] is a W3C-recommended language for expressing customised mappings from relational databases to RDF datasets. R2RML mappings create custom RDF views on top of existing relational data. The mappings themselves are RDF graphs, expressed in a structure and RDF vocabulary of choice, and encoded in Turtle syntax. An example mapping is given in *Listing* 2.1.

Listing 2.1: Example R2RML mapping

```
[]
    rr:logicalTable [ rr:tableName "CUSTOMERS" ];
    rr:subjectMap [
        rr:template "http://example.com/person/{CUSTID}";
        rr:class foaf:Person;
    ];
    rr:predicateObjectMap [
        rr:predicate foaf:name;
        rr:objectMap [ rr:column "NAME" ];
    ];
    rr:predicateObjectMap [
        rr:predicate foaf:gender;
        rr:objectMap [ rr:column "GENDER" ];
    ].
```

As we will attempt to go beyond relational databases, the existing RDB2RDF solutions will not be sufficient. **RML**[6] is a superset of R2RML which can be used to map relational databases to the RDF data model. Instead of exclusively defining table names, RML can support any reference to any source within its Logical Source (rml:LogicalSource) which extends R2RML's Logical Table (rr:LogicalTable), thereby introducing support for a broader range of data formats. An example mapping of a JSON document to an RDF view is given in *Listing* 2.2.

Listing 2.2: Example RML mapping

```
[]
    rml:logicalSource [
        rml:source "customers.json";
        rml:referenceFormulation ql:JSONPath;
        rml:iterator "$.[*].Person" ];
    rr:subjectMap [
        rr:template "http://example.com/person/{customer_id}";
```

[5] https://www.w3.org/TR/r2rml
[6] http://rml.io

```
        rr:class foaf:Person ];
    rr:predicateObjectMap [
        rr:predicate foaf:name;
        rr:objectMap [ rml:reference "name" ] ];
    rr:predicateObjectMap [
        rr:predicate foaf:gender;
        rr:objectMap [ rml:reference "gender" ] ].
```

Assuming the same data is present in the `CUSTOMERS` table and `customers.json` described above, both mappings would produce the same resulting graph. An example is shown in Listing 2.3.

Listing 2.3: Resulting RDF graph

```
ex:1234 a foaf:Person ;
  foaf:name "Jane␣Doe";
  foaf:gender "Female".
```

Although it may be seen as a data integration exercise, it is worth noting that, from an end-user perspective, a personal data inventory might not necessarily require direct/live data access, as it can also serve merely as a metadata repository. The extent to which the above solutions will be used is yet to be investigated.

2 Cataloging

It is assumed that 90% of all deployed data lakes today have entirely lost their value as they have become overwhelmed with information assets captured for uncertain use cases.[7] Both Tableau[8] and Gartner[9] refer to metadata catalogs as the most important new trend in Data Management and Analytics. Other indicators, such as DataHub.io moving away from CKAN[10], however, have also shown that traditional data cataloging solutions are no longer sufficient.

Proper semantic lifting and interlinking of the above discussed PII would also pave the way for automatic classification and cataloging (with little to no human intervention), thereby considerably minimizing the manual effort needed to maintain such information in fast-paced organisations.

Linking automatically classified and semantically annotated PII to relevant business-level definitions would also introduce context, and even allow for ad-hoc recontextualisation of data. In addition to making the process of building and enforcing policies easier, the same approach holds the potential to be extended to a more general enterprise use case involving other types of data as part of a smarter Enterprise Metadata Management solution.

The surfaced data would provide a unified and comprehensive view of all data assets across the enterprise and the means to unlock some of its *tribal knowledge*[11]. SPECIAL will investigate the scalability aspects of such a solution, but also look into advancing the state of the art in navigation and discovery.

[7] https://blogs.gartner.com/merv-adrian/2014/12/30/prediction-is-hard-especially-about-the-future/
[8] https://www.tableau.com/resource/top-10-big-data-trends-2017
[9] https://www.gartner.com/doc/3837968/data-catalogs-new-black-data
[10] http://datahub.io/docs/about
[11] https://www.isixsigma.com/dictionary/tribal-knowledge

Chapter 3

Modelling the SPECIAL Policy Log

1 The Ledger

As motivated in the previous sections, at the core of any transparency and compliance architecture is the logging of events in relation to the processing and sharing of personal data, as well as actions related to the consent provided (or revoked) by a data subject. The logs should be represented in a way that enable verification that data processors abide by the access and usage control policies that have been associated with the data based on the data subject's consent and the applicable regulations.

In *D1.3 Policy, transparency and compliance guidelines V1 - Chapter 3* - (i) we identified the main requirements for the ledger (analogous to a "log") in order to provide transparency to data subjects, e.g the *completeness*, *interoperability* and *integrity* of the log, to name but a few, (ii) we outlined the main data to be captured so that the log can be used to automatically verify compliance with access and usage control policies specified by data subjects, i.e. we envisioned to maintain a record of all data processing and sharing events, (iii) we analysed the limitations of current ledgers, and (iv) we proposed to leverage the power of RDF and Linked Data to represent the events in the ledger in a machine readable manner.

Hereinafter, we focus on providing a concrete model to represent data processing and sharing events, including the consent provided by the data subject and subsequent changes to or revocation of said consent.

To do so, we provide a new vocabulary, referred to as *SPLog* (presented in the next section) that builds upon the *policy language* ontology presented in *Deliverable D2.1 Policy Language V1* and reuses well-known vocabularies such as PROV [7] to provide provenance metadata of the log.

By employing RDF/Linked Data technologies to represent the provenance events stored in the ledger we pursue the following contributions:

- We set the basis to extend the event descriptions to cope with novel business, transparency and compliance needs, as vocabularies can be extended seamlessly.
- Events are described in semantically unambiguous terms aligned to the same taxonomies defining usage policies, hence we facilitate automatic compliance checking. The first version of the compliance checking algorithm can be found in *Deliverable D2.4*.
- We support interoperability between ledgers thanks to RDF and Linked Data principles.

A first approach to integrate several ledges can be found in Section 2.8, which will be further investigated in future versions of this deliverable.

- Events can be grouped to facilitate scalability. Section 2.6 provides a first approach, while we plan further investigation in future versions of this deliverable. Note that we provide in *Deliverable D2.4* a first big data architecture able to deal with the amount of information generated in the ledger.
- Events can be attached to an immutable record, potentially stored in a different ledger or knowledge base, to assure immutability. A discussion on potential immutable ledgers can be found in *D1.3 Policy, transparency and compliance guidelines V1*, while we discuss on using concrete immutable ledgers in *Deliverable D2.4*.
- Services providing transparency on top of the ledger can be built upon SPARQL [4] and simple inference mechanisms (as detailed in *Deliverable D2.4*).

For the sake of simplicity, and following our iterative model, we make the following assumptions in this deliverable:

- We consider a log with monotone increasing size, disregarding *erasure*. In future versions of this deliverable, we will inspect on cryptographic deletion mechanisms in order to harmonise mandatory preservation requirements and the right to erasure.
- The content of the events could potentially be described at different granularities, from categorising the content in a simple taxonomy stating the type of data, processing, etc., involved in the event, to the most fine-grained description of the actual data associated to the event (e.g. concrete location of a data subject). On the one hand, this would allow companies to have flexibility with respect to the level of detail recorded in the log. On the other hand, if actual data is stored in the log (i.e. instances), the compliance checking mechanisms may need to perform a preprocessing step to infer the actual categories (classes) that should be verified against the consent provided by data subjects. This deliverable assumes the former case, i.e. log entries store categories (classes) such that compliance checking is based on class subsumption (as detailed in *Deliverable D2.4*). Thus, we consider that actual data can be stored, linked and retrieved from an alternative data source. We will look into the need to record and do compliance checking over actual data (instances) in the future.
- The log could potentially include the formalisation of the GDPR (see *D1.3 Policy, transparency and compliance guidelines V1* presenting initial guidelines) in order to integrate, in a single system, the log of the usage policies and the regulation policies in place. We design our model considering this potential integration, hence the taxonomy of log entries is flexible enough to accommodate the formalisation of the GDPR (in essence, they could be a specific type of "*Policy Entries*" with an initially fixed validity). Nonetheless, we postpone this potential inclusion to future versions of this deliverable.
- Consents given by data subjects have a starting validity time and are defined forever unless a consent is replaced with a new consent (i.e. a consent replaces any previous consent). Updates and revocations are then implicit. Nonetheless we consider in this deliverable that companies may require to store a "*Consent Revocation*" action to simplify consent

tracking and consent versioning. In the future, we may consider further actions and associated metadata aligned with our use cases.

2 SPLog Vocabulary

This section introduces the **SPECIAL Policy Log Vocabulary (SPLog)**, a vocabulary to log data processing or sharing events (that should comply with a given privacy policy) as well as actions related to the consent provided (or revoked) by a data subject. Appendix 1 includes the formal vocabulary. We also provide concrete examples using the SPECIAL BeFit scenario of fitness tracking presented in *D1.3 Policy, transparency and compliance guidelines V1*.

2.1 Preliminaries

For designing the SPLog vocabulary we have carefully aligned with (i) the guidelines in *D1.3 Policy, transparency and compliance guidelines V1*, (ii) our policy language in *Deliverable D2.1 Policy Language V1*, (iii) the vocabulary and standard guidelines in *Deliverable 6.3 Plan for community group and standardisation contribution* and (iv) related work on log and event processing, with particular attention to the application in Linked Data scenarios.

In particular, we first followed (a) the large body of work in the Business Process Management (BPM) community that focuses on using process execution events for business process compliance monitoring [8] and process mining [11]. From this context, we borrow the following assumptions:

- We assume that a log entry contains data related to a *single process*. Thus, in our modelling, log entries are related to a process uniquely identified. For the sake of simplicity, we assume that data processing and sharing events correspond to exactly one *log entry*.
- In principle, events are instantaneous, thus they can be associated to a single timestamp. In addition, we decided to allow for grouping entries and include a duration (start and end timestamps) to favour scalability. For instance, a company may decide to insert one entry for each location gathered in the BeFit device. However this might result in scalability issues if the gathering rate is high. In contrast, the company can opt to register a log entry group for a *running activity*, hence the log entry group states that the position of the user was collected in a particular time frame.
- We integrated an optional *BPM* module in our model, in order to represent BPM information that might be present in the company and can complement the logging information. As such, each event in the log is related to a single process instance, typically called "*case*" in BPM. This in reflected in the model with a *Case* class. For example, a case could identify the daily routine of a user in our BeFit scenario. Events can be related to some activity. For example, an activity in BeFit could be "*userIsTraining*" which can be associated to several events. Nonetheless, not all logs and companies follow an organised BPM structure, hence we consider and encourage this distinction, although it is seen as optional for the model. Note that grouping the information according to BPM can help the data subject to discern about the concrete purpose of each individual action. For example, a process to perform a recommendation based on the training of data subjects can have an instance `recommendationUser1` that may consist of a set of events that *Collect* data, an event to *Analyse* data and a final event to *Recommend* something to the data subject.

- We integrated an optional *Immutable* module to represent that a log entry can be additionally linked to its representation as an immutable record, potentially stored in a different ledger or knowledge base. A discussion on immutable ledgers can be found in Deliverable D2.4.

2.2 Conventions and namespaces

The namespace for the SPECIAL Policy Log Vocabulary is `http://www.specialprivacy.eu/langs/splog#`. We write triples in this document in the Turtle RDF syntax [10] using the following namespace prefixes:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX spl: <http://www.specialprivacy.eu/langs/usage-policy#>
PREFIX splog: <http://www.specialprivacy.eu/langs/splog#>
PREFIX dct: <http://purl.org/dc/terms/>
PREFIX prov: <http://www.w3.org/ns/prov#>
PREFIX skos: <http://www.w3.org/2004/02/skos/core#>
```

The key words *MAY*, *MUST*, and *SHOULD* are to be interpreted as described in [1].

2.3 Outline of the vocabulary

Figure 3.1 depicts an overview of the vocabulary. Several concepts and properties have been defined to cover the log and its entries, detailed in Section 2.4. In addition, the description of a policy log can be complemented with two optional conceptual modules (dashed), *BPM* and *Immutable*. As stated, BPM represents the optional BPM information from the company that can be attached to events, such as activities, cases and processes. In turn, an *Immutable Record* preserves a log entry, potentially in a different ledger or knowledge base (see Deliverable D2.4 for a discussion on immutable ledgers).

2.4 Concepts

In this section we first define the main concepts concerning the SPLog vocabulary. Then, we present a practical example where we consider the SPECIAL BeFit scenario of fitness tracking, introduced in *D1.3 Policy, transparency and compliance guidelines V1*.

2.4.1 Log

A log (represented as `splog:Log`) is a collection of data that records data processing and sharing events as well as consent-related activities (assertion and revocation). The data in a log can be described as belonging to one of the following categories:

Log metadata. This is metadata that *SHOULD* describe the log as a whole, such as the label or title, the software agent(s) it belongs to, etc. Metadata is described in Section 2.7. One of the most important aspect is the processor whose service is logged. This is modelled with the `splog:processor` property (a subproperty of `prov:agent`), relating the `splog:Log` and the corresponding `splog:Processor` instance.

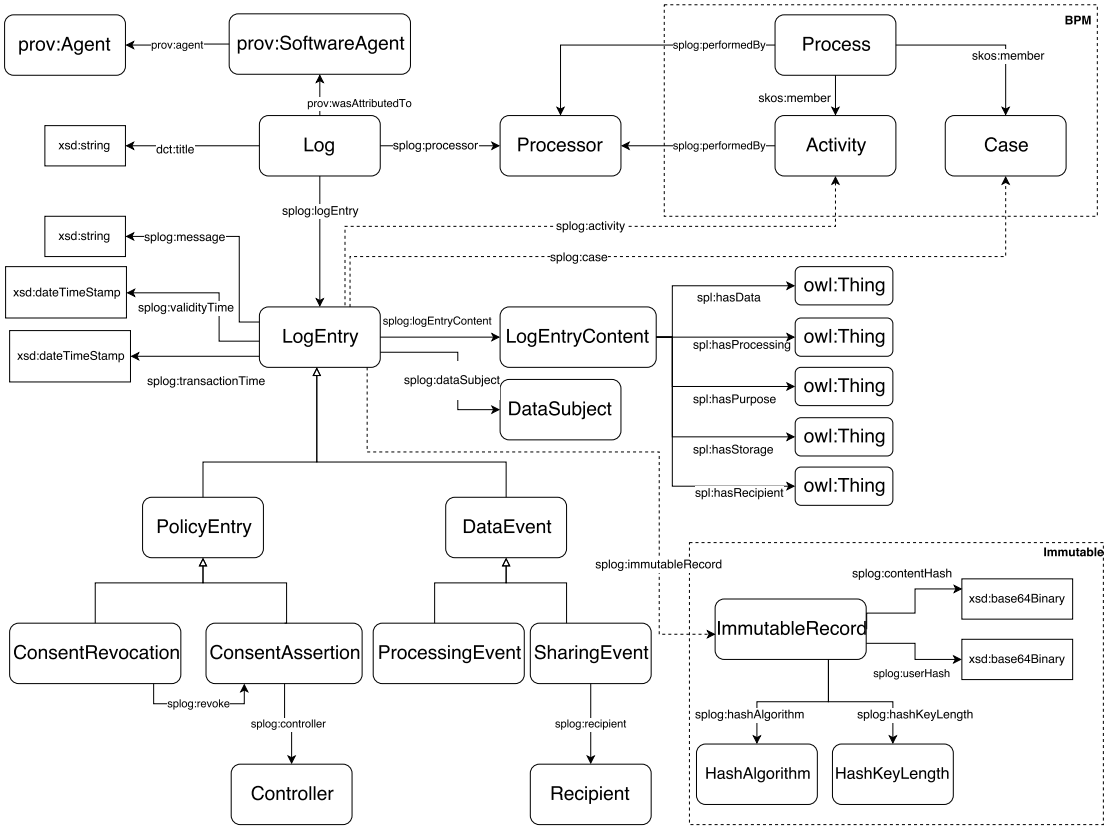

Figure 3.1: Pictorial summary of key terms and their relationship

Log Entries. This is the actual data contained in the log, represented with `splog:LogEntry`. The log *MUST* make use of the `splog:logEntry` property (a `prov:wasGeneratedBy` subproperty) to point to each entry in the log. Optionally, and for the sake of compactness, entries *MAY* be grouped into a given dimension or set of dimensions, conforming log entry groups. This is described in Section 2.6. In such case, the log can point to the groups through the specific property `splog:logEntryGroup` (a subproperty of `splog:logEntry`).

2.4.2 Log entries

Log entries contain information about processing and sharing events associated to data subjects, as well as actions related to the consent provided (or revoked) by data subjects. These different types of entries are represented in our model with a classification of log entries, i.e., a hierarchy of classes, shown in Figure 3.1. Thus, a `splog:LogEntry` has two main types (subclasses), `splog:PolicyEntry` and `splog:DataEvent`, described as follows:

`PolicyEntry`. This class reflects log entries related to policies and consent. We currently consider two subclasses, `splog:ConsentAssertion` specifying a consent provided by a data subject to a `splog:Controller` (linked with a `splog:controller` property), and `splog:ConsentRevocation`, denoting the revocation of a given consent. Note that, as stated, we assume that a consent provided by a data subject replaces any previous consent, hence consent updates are implicit. Nonetheless, companies may wish to explicitly record a revocation entry pointing to the revoked consent, thus we include this capability via the `splog:revoke` property in our model. This latter may facilitate consent tracking and consent versioning.

`DataEvent`. This class considers log entries that are actually events on the data, i.e., the aforementioned data processing and sharing events. In the case of the latter, the concrete `splog:Recipient` instances can be specified, via the `splog:recipient` property.

In turn, the data in a log entry can be described as belonging to one of the following kinds:

Log entry metadata. This is metadata that *SHOULD* describe the entry as a whole. Metadata is described in Section 2.7.

Data subjects. The log entry *SHOULD* reference the data subject(s) involved in the entry. This is specified with the `splog:dataSubject` property (a `prov:wasAssociatedWith` subproperty) pointing to the appropriate `splog:DataSubject` involved in the entry. For the sake of simplicity, we assume that an entry is related to a single data subject, but multiple data subjects *MAY* be specified using multiple `splog:dataSubject` properties. Note that in case of anonymised logs, no subject can be specified.

Content. The log entry *MUST* reference the actual data of the log. This is specified with the `splog:logEntryContent` property, which points to the appropriate instance of `splog:LogEntryContent`. This is described in Section 2.5.

Timestamps. The log entry *MUST* reference the time at which the event occurred using the `splog:validityTime` property (subproperty of `prov:atTime`). Note that this is based on the aforementioned assumption of representing instantaneous events. For the sake

of log preservation, the log entry *SHOULD* also reflect the time in which the log was recorded, using `splog:transactionTime` (a `dct:issued` subproperty).

Message. The log entry *SHOULD* reference a `splog:message` of the log representing a human-friendly text.

InmutableRecord. The entry *MAY* reference a `splog:InmutableRecord` of its contents.

Activities. The log entry *MAY* reference the concrete BPM `splog:Activity` and the BPM `splog:Case`, via the `splog:activity` and `splog:case` properties, respectively. Activities and cases are members of a `splog:Process`, specified with `skos:member`, and they can point to the involved `splog:Processor`, via `splog:performedBy`.

2.4.3 Examples

The following example provides a quick overview of how the SPECIAL Policy Log vocabulary might be used to represent a log. We make use of our BeFit scenario: we assume (i) Sue is using a wearable appliance for fitness tracking from BeFit, (ii) the application is tracking the location of Sue for *health* purposes, (iii) a new location is stored in a particular database (called *BeFitDatabaseEurope*) and reflected in the log (called *BeFitLog*). Let us also assume that the namespace for the BeFit company is `befit:` (pointing to the appropriate IRI), being `befit:Us` the main reference of the company. We first show the general log description in Listing 3.1.

Listing 3.1: Log description for BeFit devices

```
befit:BeFitLog a splog:Log;
    dct:title              "Log of BeFitDatabaseEurope"@en;
    dct:description        "This contains events on BeFitDatabaseEurope
                            tracking devices geo-located in Europe"@en;
    dct:issued             "2018-02-14"^^xsd:date;
    prov:wasAttributedTo   befit:BeFitDatabaseEurope ;
    splog:processor        befit:Us .
```

Then, we include a new entry in the log, which is a processing event (uniquely identified as `befit:entry3918`) referencing a new tracking position of Sue, shown in Listing 3.2. We assume Sue's unique identifier is `befit:Sue`, which of course, as all the log information can be kept internal to the company. The collection of the new position took place on the 3rd of January, 2018, at 13:20 (i.e. validity time) and the event was recorded few seconds later (i.e. transaction time).

Listing 3.2: A new event for Sue's BeFit device

```
befit:BeFitLog splog:event befit:entry3918 .

befit:entry3918 a splog:ProcessingEvent;
    dct:title               "Collection of new device position"@en;
    splog:dataSubject   befit:Sue ;
    dct:description         "We collected a new position of your BeFit
                             device in our database in Europe"@en;
    splog:transactionTime   "2018-01-10T13:20:50Z"^^xsd:dateTimeStamp;
    splog:validityTime      "2018-01-10T13:20:00Z"^^xsd:dateTimeStamp;
    splog:message           "Tracking position by GPS... collected!" ;
    splog:eventContent      befit:content3918 ;
    splog:inmutableRecord   befit:iRec3918 .
```

Note that, in the previous description, the log entry `befit:entry3918` is an instance of a `ProcessingEvent`, `befit:iRec3918` represents the immutable version of the event (described below), and `befit:content3918` points to the actual content of the event, defined in the following Listing 3.3.

Listing 3.3: The content of a new event for Sue's BeFit device

```
befit:content3918 a splog:LogEntryContent;
    dct:description      "This contains the data collected by a BeFit
                         device on January 2018 in Vienna, only for
                         the health purpose of the service"@en;
    spl:hasData          svd:Location;
    spl:hasProcessing    befit:SensorGathering;
    spl:hasPurpose       befit:HealthTracking;
    spl:hasStorage       [has:location svl:OurServers];
    spl:hasRecipient     [a svr:Ours].

befit:SensorGathering rdfs:subClassOf svpr:Collect .
befit:HealthTracking rdfs:subClassOf svpu:Health .
```

In turn, the immutable record can be defined as the hash of the content and the data subject, which can be kept in a different ledger or knowledge base, together with the definition of the hash algorithm. A simple example is shown in Listing 3.4.

Listing 3.4: A new event for Sue's BeFit device

```
befit:iRec3918    a splog:InmutableRecord ;
            splog:hashContent "AZ8QWE..."^^xsd:base64Binary ;
            splog:hashUser    "BHJQQ..."^^xsd:base64Binary ;
            splog:hashAlgorithm eg:hashRSA ;
            splog:hashKeyLength eg:hash2048 .
```

2.5 Log entry content

The log entry content is represented by the `splog:LogEntryContent` class, which is a type of (`rdfs:subClassOf`) the SPECIAL `spl:Authorization` defined in *Deliverable D2.1 Policy Language V1*. This way, event content and data policy authorisations can be checked for compliance. Nonetheless, note that the concept *spl:Authorization* can be confusing in this scenario as this can refer to a policy or an actual *operation* reflected in the log. Thus, we are planning to rename the `spl:Authorization` class in the new versions of the policy ontology, e.g. using `DataUsage` with the understanding that a set of `DataUsage` instances can either be policies (i.e. authorisations) or actual operations reflected in the log.

The `splog:LogEntryContent` class definition *MUST* include the five elements defined in the SPECIAL usage policy language (see *Deliverable D2.1 Policy Language V1*):

spl:hasData. It specifies the data involved in the event.

spl:hasProcessing. It specifies specifies how is data processed.

spl:hasPurpose. It specifies the purpose of the data processing.

spl:hasStorage. It specifies where and for how long is the data stored.

spl:hasRecipient. It specifies potential disclosures to other recipients, including third parties.

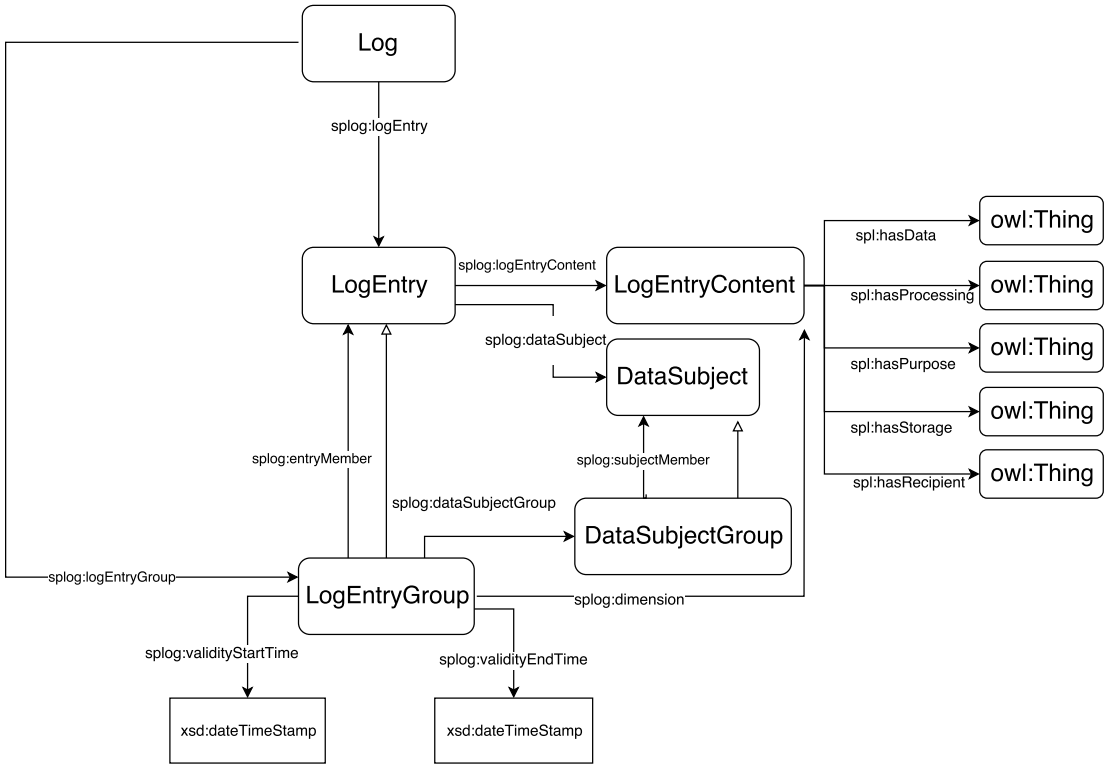

Figure 3.2: Pictorial summary of log entry grouping

Further information on compliance checking between the log entry content and the consent provided by the data subject can be found in Deliverable D2.4.

2.6 Grouping log entries

Log entries, and in particular data processing and sharing events, are meant to provide fine-grained descriptions, typically concerning a single data subject and action (i.e. a single data processing and sharing event). Thus, in those scenarios where there exists a continuous flow of information, such as the envisioned big data application, log processing can quickly suffer from scalability issues when it is needed to serve both transparency and compliance purposes. The scalable, big data infrastructure proposed in *Deliverable D2.4* is aimed at coping with such scenarios. In addition, we envision grouping mechanisms to group, slice and dice log entries in order to support presentation and processing. In the following, we present the `splog:LogEntryGroup` concept, which will be further detailed in future versions of this deliverable.

A log entry group is a subclass of a log entry, containing information about one or more log entries. For instance, in our BeFit use case, a log entry group could be used to represent (as a single entry) the collection of data during a running activity of a data subject in BeFit. Another scenario regards grouping information for several data subjects, e.g. to group the act of providing a recommendation to several subjects. Figure 3.2 shows and overview of the main components for grouping.

The data in a group can be described as belonging to one of the following kinds:

Log entry group metadata. This is metadata that describes the log group as a whole. Metadata is described in Section 2.7.

Timestamps. The entry *MAY* reference the time interval considered in the group, using the `splog:validityStartTime` and `splog:validityEndTime` properties (subproperties of `prov:startedAtTime` and `prov:endedAtTime`, respectively), denoting the validity time. Similarly to log entries, a log entry group SHOULD also reflect the time in which the entry was recorded using `splog:transactionTime` (a `dct:issued` subproperty).

Dimensions. The group *MUST* reference the component(s) it groups. This is specified with the `splog:dimension` property (a subproperty of `splog:logEntryContent`), pointing to a particular `splog:LogEntryContent`.

Data subject. The group *MAY* reference the data subject(s) it groups, using the property `splog:dataSubjectGroup` (`prov:wasAssociatedWith` subproperty). This property points to a `splog:DataSubjectGroup` instance that groups all the data subject members in the group via `splog:subjectMember` (a `skos:member` subproperty).

Entries. The group *MAY* point to the particular entries included in the group through the `splog:entryMember` property (a `skos:member` subproperty). This can serve the traceability requirement identified in *D1.3 Policy, transparency and compliance guidelines V1 - Chapter 3* -, stating that it should be possible to know about any previous processing of the data and link events in a manner that supports traceability of processing.

The following example in Listing 3.5 shows a log grouping the category of recommendations given to Sue, John and Rose during a month.

Listing 3.5: A grouping example merging all recommendations given in a month

```
befit:BeFitLog a splog:Log ;
    splog:logEntryGroup befit:recommendationsJanuary2018 .

befit:recommendationsJanuary2018 a splog:logEntryGroup
    splog:transactionTime    "2018-02-01T00:05:00Z"^^xsd:dateTimeStamp;
    splog:validityTime       "2018-01-31T23:59:59Z"^^xsd:dateTimeStamp;
    splog:dataSubjectGroup   befit:basicSubjectGroup;
    splog:dimension          befit:templateOfferRecommendation .

befit:basicSubjectGroup splog:member befit:Sue, befit:John, befit:Rose.

befit:templateOfferRecommendation a splog:LogEntryContent ;
    spl:hasData              befit:OfferRecommendation;
    spl:hasProcessing        befit:MonthlyDataAnalysis;
    spl:hasPurpose           befit:MonthlyOffersRecommendation;
    spl:hasStorage           [has:location svl:OurServers];
    spl:hasRecipient         [a svr:Ours].

befit:OfferRecommendation rdfs:subClassOf svd:Location;
    rdfs:comment  "We recommended you an offer at the end of the month
                   based on the location of your device during the
```

```
                given month. The concrete offer is not stored in
                this log" .

befit:MonthlyDataAnalysis rdfs:subClassOf svpr:Analyze .
befit:MonthlyOffersRecommendation rdfs:subClassOf
   befit:RecommendationActivity .
befit:RecommendationActivity rdfs:subClassOf svpu:Marketing .
```

2.7 Log metadata

Logs, log entries and log entry groups *SHOULD* be marked up with metadata to support presentation and processing. Dublin Core Terms [3] *SHOULD* be used for representing the key metadata annotations commonly needed for Logs. The recommend minimum core set of metadata terms is:

- `dct:title` - may be same as `rdfs:label`
- `dct:description` - may be same as `rdfs:comment`
- `dct:issued` and `dct:modified` - may specify additional times.

Additional metadata terms can be used for describing policy logs, which will be discussed in the upcoming *W3C workshop on Data Privacy Controls and Vocabularies*[1] together with a new evolution of the policy language.

2.8 Provenance information

In the log model, we assume that the description of entries coming from different systems can be merged and integrated together in a single store, which will potentially serve transparency and compliance mechanisms.

In certain scenarios, named graphs can be used to encapsulate logs before integrating entries coming from different subsystems. For example, let us assume a gym company "ViennaGym", referred to with the namespace `viennagym`, makes offers to Sue based on a mutual sharing policy with BeFit. Listing 3.6 builds upon the previous gathering event (see Listing 3.2) and shows the integration with a marketing event from ViennaGym providing offers to Sue. First, the data item is gathered by BeFit (previous example), then it is shared between BeFit and ViennaGym, and finally this latter uses the data to provide marketing advertising. These series of events are encapsulated in three graphs `befit:tracking`, `befit:sharing`, and `viennagym:marketing` respectively. We make use of the TriG [2] syntax to extend Turtle with named graphs.

[1] https://www.w3.org/2018/vocabws/

Listing 3.6: Example of integrating several events for Sue's BeFit device using provenance information in named graphs

```
# The default graph may include metadata about the graphs

befit:tracking prov:agent befit:Us .
befit:sharing prov:agent befit:Us .
viennagym:marketing prov:agent viennagym:Us .

# The following graph encodes information gathered from BeFit devices
befit:tracking{

  befit:entry3918 a splog:ProcessingEvent;
     prov:wasAssociatedWith befit:Sue ;
     splog:transactionTime  "2018-01-10T13:20:50Z"^^xsd:dateTimeStamp;
     splog:validityTime     "2018-01-10T13:20:00Z"^^xsd:dateTimeStamp;
     splog:message          "Tracking position by GPS.. collected!" ;
     # ... other metadata ...
     splog:content          befit:content3918 .

  befit:content3918 a splog:LogEntryContent;
     spl:hasData          svd:Location;
     spl:hasProcessing    befit:SensorGathering;
     spl:hasPurpose       befit:HealthTracking;
     spl:hasStorage       [has:location svl:OurServers];
     spl:hasRecipient     [a svr:Ours].

  # ... other descriptions ...

}

# This graph encodes sharing events between BeFit and ViennaGym
befit:sharing{

  befit:entry4253 a splog:SharingEvent;
     prov:wasAssociatedWith befit:Sue ;
     splog:transactionTime  "2018-01-15T09:02:30Z"^^xsd:dateTimeStamp;
     splog:validityTime     "2018-01-15T09:00:00Z"^^xsd:dateTimeStamp;
     splog:message          "Sharing GPS positions with a partner" ;
     # ... other metadata ...
     splog:recipient        viennagym:Us ;
     splog:content          befit:content4253 .

  befit:content4253 a splog:LogEntryContent;
     spl:hasData          svd:Location;
     spl:hasProcessing    befit:SecureTransferPartner;
     spl:hasPurpose       befit:BefitpartnerRecommendation;
     spl:hasStorage       [has:location svl:OurServers];
     spl:hasRecipient     viennagym:Company.


  befit:SecureTransferPartner rdfs:subClassOf svpr:Transfer .
  befit:PartnerRecommendation rdfs:subClassOf
     befit:RecommendationActivity .
  befit:RecommendationActivity rdfs:subClassOf svpu:Marketing .
  viennagym:Company rdfs:subClassOf spl:AnyRecipient .
}
```

```
# This graph encodes the marketing information of Sue by ViennaGym
viennagym:marketing{

viennagym:entry1111 a splog:ProcessingEvent;
      prov:wasAssociatedWith befit:Sue ;
      splog:transactionTime  "2018-01-27T13:00:30Z"^^xsd:dateTimeStamp;
      splog:validityTime     "2018-01-27T13:00:00Z"^^xsd:dateTimeStamp;
      splog:message          "Send offer of our gym!" ;
      # ... other metadata ...
      splog:content          viennagym:marketing6590 .

 viennagym:marketing6590 a splog:LogEntryContent;
       spl:hasData          svd:Location;
       spl:hasProcessing    viennagym:Analysis;
       spl:hasPurpose       viennagym:GymRecommendation;
       spl:hasStorage       [has:location svl:OurServers];
       spl:hasRecipient     [a svr:Ours].

 viennagym:Analysis rdfs:subClassOf svpr:Analyze .
 viennagym:GymRecommendation rdfs:subClassOf svpu:Marketing .

}
```

Chapter 4

Appendix

1 The SPECIAL Policy Log Vocabulary

```
@prefix : <http://www.specialprivacy.eu/langs/splog#> .
@prefix dct: <http://purl.org/dc/terms/> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix spl: <http://www.specialprivacy.eu/langs/usage-policy#> .
@prefix xml: <http://www.w3.org/XML/1998/namespace> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix prov: <http://www.w3.org/ns/prov#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .

<http://www.specialprivacy.eu/langs/splog> a owl:Ontology ;
rdfs:seeAlso "https://aic.ai.wu.ac.at/qadlod/policyLog/" ;
owl:versionInfo "0.3"@en .
#
#
# ////////////////////////////////////////////////////////////////
# #
# #    Object Properties
# #
# ////////////////////////////////////////////////////////////////
#
#
# http://www.specialprivacy.eu/langs/splog#activity

:activity a owl:ObjectProperty ;
rdfs:domain :LogEntry ;
rdfs:range :Activity .
#
# http://www.specialprivacy.eu/langs/splog#case

:case a owl:ObjectProperty ;
rdfs:subPropertyOf owl:topObjectProperty ;
rdfs:domain :LogEntry ;
rdfs:range :Case .
#
# http://www.specialprivacy.eu/langs/splog#controller

:controller a owl:ObjectProperty ;
rdfs:subPropertyOf prov:agent ;
rdfs:domain :ConsentAssertion ;
rdfs:range :Controller .
#
# http://www.specialprivacy.eu/langs/splog#dataSubject

:dataSubject a owl:ObjectProperty ;
rdfs:subPropertyOf prov:wasAssociatedWith ;
rdfs:domain :LogEntry ;
rdfs:range :DataSubject .
#
# http://www.specialprivacy.eu/langs/splog#dataSubjectGroup
```

```
:dataSubjectGroup a owl:ObjectProperty ;
rdfs:subPropertyOf prov:wasAssociatedWith ;
rdfs:domain :DataSubjectGroup ;
rdfs:range :DataSubject .
#
# http://www.specialprivacy.eu/langs/splog#dimension

:dimension a owl:ObjectProperty ;
rdfs:subPropertyOf :logEntryContent ;
rdfs:domain :LogEntryGroup ;
rdfs:range :LogEntryContent .
#
# http://www.specialprivacy.eu/langs/splog#entryMember

:entryMember a owl:ObjectProperty ;
rdfs:subPropertyOf <http://www.w3.org/2004/02/skos/core#member> ;
rdfs:domain :LogEntryGroup ;
rdfs:range :LogEntry .
#
# http://www.specialprivacy.eu/langs/splog#hashAlgorithm

:hashAlgorithm a owl:ObjectProperty ;
rdfs:domain :ImmutableRecord ;
rdfs:range :HashAlgorithm .
#
# http://www.specialprivacy.eu/langs/splog#hashKeyLength

:hashKeyLength a owl:ObjectProperty ;
rdfs:subPropertyOf owl:topObjectProperty ;
rdfs:domain :ImmutableRecord ;
rdfs:range :HashKeyLength .
#
# http://www.specialprivacy.eu/langs/splog#immutableRecord

:immutableRecord a owl:ObjectProperty ;
rdfs:subPropertyOf prov:wasGeneratedBy ;
rdfs:domain :LogEntry ;
rdfs:range :ImmutableRecord .
#
# http://www.specialprivacy.eu/langs/splog#logEntry

:logEntry a owl:ObjectProperty ;
rdfs:subPropertyOf prov:wasGeneratedBy ;
rdfs:domain :Log ;
rdfs:range :LogEntry .
#
# http://www.specialprivacy.eu/langs/splog#logEntryContent

:logEntryContent a owl:ObjectProperty ;
rdfs:domain :LogEntry ;
rdfs:range :LogEntryContent ;
rdfs:comment "Associates the Event with its content" .
```

```
#
# http://www.specialprivacy.eu/langs/splog#logEntryGroup

:logEntryGroup a owl:ObjectProperty ;
rdfs:subPropertyOf prov:wasGeneratedBy ;
rdfs:domain :Log ;
rdfs:range :LogEntryGroup .
#
# http://www.specialprivacy.eu/langs/splog#performedBy

:performedBy a owl:ObjectProperty ;
rdfs:domain :Activity ;
rdfs:range prov:Agent .
#
# http://www.specialprivacy.eu/langs/splog#processor

:processor a owl:ObjectProperty ;
rdfs:subPropertyOf prov:agent ;
rdfs:domain :Log ;
rdfs:range :Processor .
#
# http://www.specialprivacy.eu/langs/splog#recipient

:recipient a owl:ObjectProperty ;
rdfs:domain :SharingEvent ;
rdfs:range :Recipient .
#
# http://www.specialprivacy.eu/langs/splog#revoke

:revoke a owl:ObjectProperty ;
rdfs:domain :ConsentRevocation ;
rdfs:range :ConsentAssertion .
#
# http://www.specialprivacy.eu/langs/splog#subjectMember

:subjectMember a owl:ObjectProperty ;
rdfs:subPropertyOf <http://www.w3.org/2004/02/skos/core#member> .
#
# http://www.w3.org/2004/02/skos/core#member

<http://www.w3.org/2004/02/skos/core#member> a owl:ObjectProperty .
#
# http://www.w3.org/ns/prov#agent

prov:agent a owl:ObjectProperty .
#
# http://www.w3.org/ns/prov#wasAssociatedWith

prov:wasAssociatedWith a owl:ObjectProperty .
#
# http://www.w3.org/ns/prov#wasGeneratedBy
```

```
prov:wasGeneratedBy a owl:ObjectProperty .
#
#
#
# ################################################################
# #
# #    Data properties
# #
# ################################################################
#
#
# http://purl.org/dc/terms/issued

dct:issued a owl:DatatypeProperty .
#
# http://www.specialprivacy.eu/langs/splog#contentHash

:contentHash a owl:DatatypeProperty ;
rdfs:domain :ImmutableRecord ;
rdfs:range xsd:base64Binary .
#
# http://www.specialprivacy.eu/langs/splog#message

:message a owl:DatatypeProperty ;
rdfs:domain :LogEntry ;
rdfs:range xsd:string .
#
# http://www.specialprivacy.eu/langs/splog#transactionTime

:transactionTime a owl:DatatypeProperty ;
rdfs:subPropertyOf dct:issued ;
rdfs:domain :LogEntry ;
rdfs:range xsd:dateTimeStamp .
#
# http://www.specialprivacy.eu/langs/splog#userHash

:userHash a owl:DatatypeProperty ;
rdfs:domain :ImmutableRecord ;
rdfs:range xsd:base64Binary .
#
# http://www.specialprivacy.eu/langs/splog#validityEndTime

:validityEndTime a owl:DatatypeProperty ;
rdfs:domain :LogEntryGroup ;
rdfs:range xsd:dateTimeStamp .
#
# http://www.specialprivacy.eu/langs/splog#validityStartTime

:validityStartTime a owl:DatatypeProperty ;
rdfs:domain :LogEntryGroup ;
rdfs:range xsd:dateTimeStamp .
#
```

```
# http://www.specialprivacy.eu/langs/splog#validityTime

:validityTime a owl:DatatypeProperty ;
rdfs:subPropertyOf prov:atTime ;
rdfs:domain :LogEntry ;
rdfs:range xsd:dateTimeStamp .
#
# http://www.w3.org/ns/prov#atTime

prov:atTime a owl:DatatypeProperty .
#
#
#
# ////////////////////////////////////////////////////////////////
# #
# #    Classes
# #
# ////////////////////////////////////////////////////////////////
#
#
# http://www.specialprivacy.eu/langs/splog#Activity

:Activity a owl:Class ;
rdfs:comment "a BPM activity"@en ;
rdfs:label "Activity"@en .
#
# http://www.specialprivacy.eu/langs/splog#Case

:Case a owl:Class ;
rdfs:comment "a BPM case"@en ;
rdfs:label "Case"@en .
#
# http://www.specialprivacy.eu/langs/splog#ConsentAssertion

:ConsentAssertion a owl:Class ;
rdfs:subClassOf :PolicyEntry ;
rdfs:comment "A consent provided by a data subject"@en ;
rdfs:label "Consent Assertion"@en .
#
# http://www.specialprivacy.eu/langs/splog#ConsentRevocation

:ConsentRevocation a owl:Class ;
rdfs:subClassOf :PolicyEntry ;
rdfs:comment "The revocation of a given consent"@en ;
rdfs:label "Consent Revocation"@en .
#
# http://www.specialprivacy.eu/langs/splog#Controller

:Controller a owl:Class ;
rdfs:subClassOf prov:Agent ;
rdfs:comment "Controller as defined by Art. 4 (7) of the GDPR"@en ;
rdfs:label "Controller"@en .
```

```
#
# http://www.specialprivacy.eu/langs/splog#DataEvent

:DataEvent a owl:Class ;
rdfs:subClassOf :LogEntry ;
rdfs:comment "Log entries that are actually events on the data, such
   as data processing and sharing events"@en ;
rdfs:label "Data Event"@en .
#
# http://www.specialprivacy.eu/langs/splog#DataSubject

:DataSubject a owl:Class ;
rdfs:subClassOf prov:Agent ;
rdfs:comment "Natural person as per Art. 4 (1) of the GDPR"@en ;
rdfs:label "Data Subject"@en .
#
# http://www.specialprivacy.eu/langs/splog#DataSubjectGroup

:DataSubjectGroup a owl:Class ;
rdfs:subClassOf :DataSubject ;
rdfs:comment "A goup of one or more data subjects"@en ;
rdfs:label "Data Subject Group"@en .
#
# http://www.specialprivacy.eu/langs/splog#HashAlgorithm

:HashAlgorithm a owl:Class ;
rdfs:comment "Defines an algorithm for hashing"@en ;
rdfs:label "Hash Algorithm"@en .
#
# http://www.specialprivacy.eu/langs/splog#HashKeyLength

:HashKeyLength a owl:Class ;
rdfs:comment "Defines the key length of a Hash Algorithm"@en ;
rdfs:label "Hash Key Length"@en .
#
# http://www.specialprivacy.eu/langs/splog#ImmutableRecord

:ImmutableRecord a owl:Class ;
rdfs:subClassOf prov:Activity ;
rdfs:comment "the immutable record of an event"@en ;
rdfs:label "Immutable Record"@en .
#
# http://www.specialprivacy.eu/langs/splog#Log

:Log a owl:Class ;
rdfs:subClassOf prov:Entity ;
rdfs:comment "A Log is a collection of data that records data
    processing and sharing events as well as consent-related
    activities (acquisition and revocation)"@en ;
rdfs:label "Log"@en .
#
# http://www.specialprivacy.eu/langs/splog#LogEntry
```

```
:LogEntry a owl:Class ;
rdfs:subClassOf prov:Activity ;
rdfs:comment "A log entry contains information about a processing and
sharing  event associated to a data subject, as well as actions
related  to the consent provided (or revoked) by a data subject"@en ;
rdfs:label "Log Entry"@en .
#
# http://www.specialprivacy.eu/langs/splog#LogEntryContent

:LogEntryContent a owl:Class ;
rdfs:subClassOf spl:Authorization ;
rdfs:comment "The content of a log entry in terms of the data involved,
how is data processed, the purpose of the process, where and for how
long is the data stored and potential disclosures to third parties"@en ;
rdfs:label "Log Entry Content"@en .
#
# http://www.specialprivacy.eu/langs/splog#LogEntryGroup

:LogEntryGroup a owl:Class ;
rdfs:subClassOf :LogEntry ;
rdfs:comment "a log entry group contains information about one or more
log  entries"@en ;
rdfs:label "Log Entry Group"@en .
#
# http://www.specialprivacy.eu/langs/splog#PolicyEntry

:PolicyEntry a owl:Class ;
rdfs:subClassOf :LogEntry ;
rdfs:comment "Log entries related to policies and consent"@en ;
rdfs:label "Policy Entry"@en .
#
# http://www.specialprivacy.eu/langs/splog#Process

:Process a owl:Class ;
rdfs:comment "a BPM process"@en ;
rdfs:label "Process"@en .
#
# http://www.specialprivacy.eu/langs/splog#ProcessingEvent

:ProcessingEvent a owl:Class ;
rdfs:subClassOf :DataEvent ;
rdfs:comment "A data processing event"@en ;
rdfs:label "Data Processing Event"@en .
#
# http://www.specialprivacy.eu/langs/splog#Processor

:Processor a owl:Class ;
rdfs:subClassOf prov:Agent ;
rdfs:comment "Processor as defined by Art. 4 (8) of the GDPR"@en ;
rdfs:label "Processor"@en .
#
```

```
# http://www.specialprivacy.eu/langs/splog#Recipient

:Recipient a owl:Class ;
rdfs:subClassOf prov:Agent ;
rdfs:comment "recipient as defined by Art. 4 (9) of the GDPR"@en ;
rdfs:label "Recipient"@en .
#
# http://www.specialprivacy.eu/langs/splog#SharingEvent

:SharingEvent a owl:Class ;
rdfs:subClassOf :DataEvent ;
rdfs:comment "A data sharing event"@en ;
rdfs:label "Data Sharing Event"@en .
#
# http://www.specialprivacy.eu/langs/usage-policy#Authorization

spl:Authorization a owl:Class .
#
# http://www.w3.org/ns/prov#Activity

prov:Activity a owl:Class .
#
# http://www.w3.org/ns/prov#Agent

prov:Agent a owl:Class .
#
# http://www.w3.org/ns/prov#Entity

prov:Entity a owl:Class .
```

Bibliography

[1] S. Bradner. Key words for use in rfcs to indicate requirement levels, 1997. URL https://tools.ietf.org/html/rfc2119.

[2] G. Carothers and A. Seaborne. Rdf 1.1 turtle: Rdf dataset language. *W3C Recommendation, February*, 2014.

[3] Dublin Core metadata initiative. Dcmi metadata terms, 2012. URL http://dublincore.org/documents/dcmi-terms/.

[4] S. H. Garlik, A. Seaborne, and E. PrudâĂŹhommeaux. SPARQL 1.1 Query Language, W3C Recommendation, 2013. URL https://www.w3.org/TR/sparql11-query/.

[5] L. Geng, L. Korba, X. Wang, Y. Wang, H. Liu, and Y. You. Using data mining methods to predict personally identifiable information in emails. In C. Tang, C. X. Ling, X. Zhou, N. J. Cercone, and X. Li, editors, *Advanced Data Mining and Applications*, pages 272–281, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg. ISBN 978-3-540-88192-6.

[6] L. Korba, Y. Wang, L. Geng, R. Song, G. Yee, A. S. Patrick, S. Buffett, H. Liu, and Y. You. Private data discovery for privacy compliance in collaborative environments. In Y. Luo, editor, *Cooperative Design, Visualization, and Engineering*, pages 142–150, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg. ISBN 978-3-540-88011-0.

[7] T. Lebo, S. Sahoo, and D. McGuinness. Prov-o: The prov ontology. *W3C Recommendation, April*, 2013.

[8] L. T. Ly, F. M. Maggi, M. Montali, S. Rinderle-Ma, and W. M. van der Aalst. Compliance monitoring in business processes: Functionalities, application, and tool-support. *Information systems*, 54:209–234, 2015.

[9] V. Nguyen, O. Bodenreider, and A. Sheth. Don't like rdf reification?: Making statements about statements using singleton property. In *Proceedings of the 23rd International Conference on World Wide Web*, WWW '14, pages 759–770, New York, NY, USA, 2014. ACM. ISBN 978-1-4503-2744-2. doi: 10.1145/2566486.2567973. URL http://doi.acm.org/10.1145/2566486.2567973.

[10] E. Prud'hommeaux and G. Carothers. Rdf 1.1 turtle: Terse rdf triple language. *W3C Recommendation, February*, 2014.

[11] W. M. Van der Aalst. Process mining. In *Process Mining*, pages 95–123. Springer, 2011.