A Provenance Model for Quantified Self Data

Andreas Schreiber(✉)

Distributed Systems and Component Software, German Aerospace Center (DLR),
Linder Höhe, 51147 Cologne, Germany
Andreas.Schreiber@dlr.de
http://www.dlr.de/sc

Abstract. Quantified Self became popular in recent years. People are tracking themselves with Wearables, smartphone apps, or desktop applications. They collect, process and store huge amounts of personal data for medical and other reasons. Due to the complexity of different data sources, apps, and cloud services, it is hard to follow the data flow and to have trust in data integrity and safety. We present a solution that helps to get insight in Quantified Self data flows and to answer questions related to data security. We provide a provenance model for Quantified Self data based on the W3C standard PROV. Using that model, developers and users can record provenance of Quantified Self apps and services with a standardized notation. We show the feasibility of the presented provenance model with a small workflow using steps data from Fitbit fitness tracker.

Keywords: Provenance · Quantified self · Personal informatics · Trust · Ontology · PROV

1 Introduction

In almost any medical visit nowadays it is common to collect additional data of a patient in form of a medical history or raw data such as weight, height and blood pressure. The data gives doctors additional knowledge about your health status, possible treatments or signs of diseases.

A self-surveillance of personal data is called Quantified Self (QS) [2,5,18]. It is a recently upcoming movement that describes a community of people (users) who record and analyze data about themselves for medical reasons, self-improvement, technological interests, or other reasons. The user collects various types of data related to him, to get a better understanding of himself. The main differences between the data collected in a medical visit and recorded by the user are, that the data recorded by the user can cover a much wider area of interest (e.g., calories consumed or distance walked) and additionally it is recorded in a much shorter period, enabling precisely analytics, that result in more reliable diagnostics than in any medical history collected by the doctor.

The goals of the community are supported by software developers and manufacturers that are providing applications and devices for the user that are easy

M. Antona and C. Stephanidis (Eds.): UAHCI 2016, Part I, LNCS 9737, pp. 382–393, 2016.
DOI: 10.1007/978-3-319-40250-5_37

to use and mostly interoperable with each other, so that the user can create a flexible chain of different devices and programs that are analyzing his data and showing him easy understandable results. But, for example, questions like *"Can I trust the developer with my data?"* or *"How dangerous can it be quantifying and optimizing myself without proper medical support?"* are now hard to answer by users.

To answer those questions and to understand how the data of the user is created, manipulated and processed, the provenance [15] of that data can be recorded and analyzed. Provenance has many synonyms like lineage, genealogy or pedigree that results from different domains in which they were elaborated separately until a few years ago.

By capturing and storing provenance self-trackers could answer questions regarding the creation process of his data, security aspects, and even determining the trustfulness of his diagnostic results (Sect. 3.1). Like any database management system, a provenance system needs an underlying data model (provenance data model) that defines the data that will be captured in QS workflows.

Our main contributions are an ontology (Sect. 3.2) and a provenance model for quantified self workflows (Sect. 3.3). It gives a starting point for the exploration of provenance in QS and related fields, such as telemedicine. We show the feasibility of the provenance model by an example where we record the provenance of simple process. We query steps data via the Fitbit API, clean the data, and visualize the data (Sect. 4).

2 Provenance of Electronic Data

The definition of *provenance* is: *"Provenance is a record that describes the people, institutions, entities, and activities involved in producing, influencing, or delivering a piece of data or a thing. In particular, the provenance of information is crucial in deciding whether information is to be trusted, how it should be integrated with other diverse information sources, and how to give credit to its originators when reusing it. In an open and inclusive environment such as the Web, where users find information that is often contradictory or questionable, provenance can help those users to make trust judgments* [16]*"*.

With the previous definition, World Wide Web Consortium (W3C) started in 2011 and finalized in 2013 the generic provenance model PROV-DM [16] and an ontology PROV-O [12], inspired by various different approaches [14], that is adaptable to any domain. The general provenance model can be seen as a property graph with three different types of nodes: *Entities*, *Activities*, and *Agents*. Entities represent physical (e.g., a scale), digital (e.g., data), conceptual (e.g., a plan), or any other kinds of objects. An activity is a process that uses or generates entities and that can be associated with an agent, meaning that the agent is responsible for the activity.

Provenance is being recorded during runtime of a process. To make QS workflows provenance-aware requires to gather information that is required by the provenance model. This information is stored in a provenance database or provenance store. For example, ProvStore [8] is publicly available provenance store.

Large provenance graphs of long running real world workflows are stored in scalable databases (e.g., Neo4j [20]). For analyzing provenance data, visualization is a feasible method. Several solutions to visualize provenance exist. For example, publicly web-based tools such as PROV-O-Viz [4], desktop tools such as VisTrails [1], or numerous other graph visualization tools exist.

3 Quantified Self Provenance Model

One of the main tasks of provenance is to answer questions regarding the process that led to a specific piece of data, its corresponding data and agents that were involved in the progress. Since provenance data could be anything related to the origination process, some sort of filter is needed, so that a provenance system can focus its resources into recording specific data, rather than simple audit everything possible.

In data modeling such a focus is usually described with use cases. Because one of the tasks of provenance is answering questions, it is obvious that for this case, to describe the focus through questions which are of interest to be answered. These questions can be modified or extended during the modeling process, because the interest on which they are based could change during a project's lifetime.

3.1 Provenance Questions for Quantified Self

In this section, ten questions will be presented and explained regarding their importance and rationale of them. These questions are just an excerpt of the many possible provenance questions in this domain, but enough to give an overview about its feasibility. With additional questions, the resulting provenance model may need to be extended too.

The following questions were formalized from the position and interest of a regular user [9]. To keep the generalized QS provenance model simple and reduce the complexity of the provenance system, mostly very general questions were considered.

Entity Focused Questions

– *What data about the user were created during the activity X?* Sometimes the user would want to know how much data of him was collected during an activity. This question aims to reveal all data generated during a specific activity.
– *What data about the user were automatically generated?* This question is a specialized form of the previous one. Since all data during an activity can be classified into automatically and manually generated data, this question sets a focus on the first one.
– *What data about the user were derived from manual input?* Although the user might know which data he entered, this question aims to reveal the data, which was derived from his manual input. Sometimes it's not clear to the user, what others could spy out with that data.

Activity Focused Questions

- *Which activities support visualization of the user's data?* In some complex cases, there can exist possibilities to use software or hardware in ways that the user did not know. This question aims to reveal all possibilities on visualizing the user's data.
- *In which activities can the user input data?* Like the previous question, this question aims on possible activities on hardware or software that the user is not aware of. In this case he could want to know which devices, applications and web services support manual data input.
- *What processes are communicating data?* This general question would reveal the complete QS workflow to the user. Depending on the effort a user takes in quantifying himself, he might loose the overview of his QS workflow. By answering this question, the user could easily keep the overview of it.
- *Which activity generated the origin of data X?* The final activity focused question more complex. Some users might want to know how a specific entity came into existence, especially where the origin data came from.

Agent Focused Questions

- *What parties were involved in generating data X?* This question, due to agent-focused view, aims like many questions of this view, a security or trustiness aspect of the QS workflow. Often the user shares data through applications and web services and view the visualized results without knowing who processed his data. This question shall reveal all parties that take responsibility in generating a specific piece of data.
- *What parties received access on data X?* Although the user may know who processed his data, it is more crucial to have particular access on a specific data, reusing it several times. This questions targets to reveal all security issues which the user might be not aware of.
- *Can other parties see user's data X?* In some applications and web services, the user can grant access of his data to other people (e.g., for competitive reasons). To get an overview, which users he had given access to a specific piece of data, is this questions target.

3.2 Quantified Self Ontology

Capturing provenance with a generic data model could be implemented easily, but the missing semantics would constraint the information gain of such a provenance system. Due to this, we developed a QS ontology. Based on Noy's and McGuinness' ontology creation guide [17], it allows inferring semantics onto PROV-DM types and relations.

Our ontology is directly related to PROV-DM with a similar hierarchy and labels (Fig. 1). Since it infers semantics on an already defined data structure, our ontology is structured into three domains: *Agents*, *Activities*, and *UserData*.

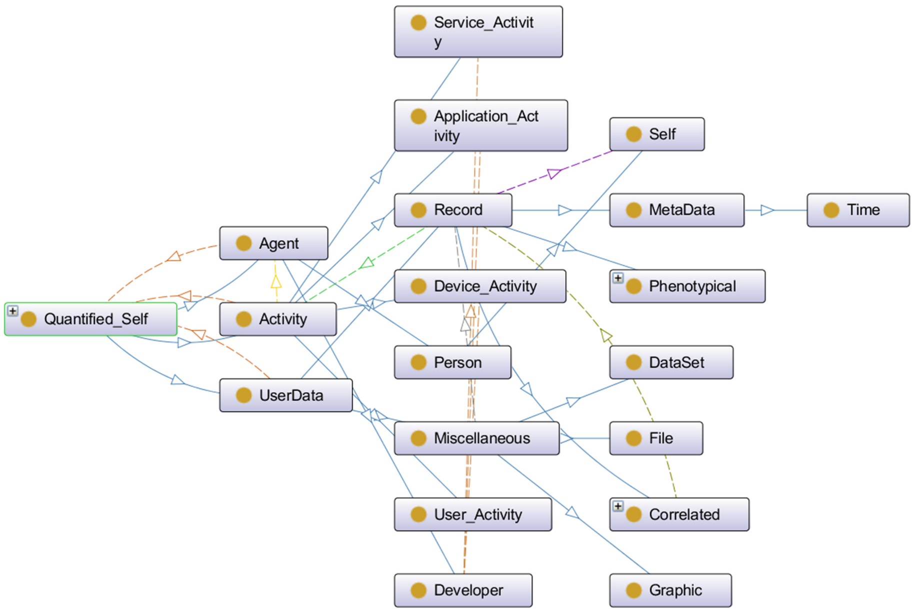

Fig. 1. Quantified self ontology.

Each domain extends PROV-DM: *Agents* ↦ *Agents*, *Activities* ↦ *Activities*, and *Entities* ↦ *UserData*.

The extensions to PROV-DM's *Agent* class are straightforward. We created a *Developer* as a subclass of *Agent*. *Developer* can be both, a *Person* or *Organization*. In the QS domain, a *Developer* produces software or hardware for users. The difference if he is a person or organization is irrelevant. Additionally, in QS the user signifies semantically more than any normal person and thus we added the class *Self* as a subclass of PROV-DM's *Person*.

We added few more subclasses to *Activity*: *User-*, *Device-*, *Application-* and *Service-Activity*. Although these activities could be structured into many fine granulated subclasses, we decided to keep the ontology on an abstract level at first and extend it if needed.

The *Entity* class had the most new semantics added. *UserData* is a subclass of *Entity*, which represents all data generated or related to the user. Since in QS various forms of data exist, we defined two subclasses: *Miscellaneous*, which represents all forms of data that are no original measured or recorded data (e.g., *Graphics*, *DataSets* or abstract *Files*), and *Record*, which represents real original (raw) data set with *Integer*'s, *Float*'s, *Double*'s, or *String*'s.

3.3 Provenance Model

Based on a requirements study of QS workflows and analysis of documentation from breakout sessions at QS Conferences (such as the QSEU14 Breakout

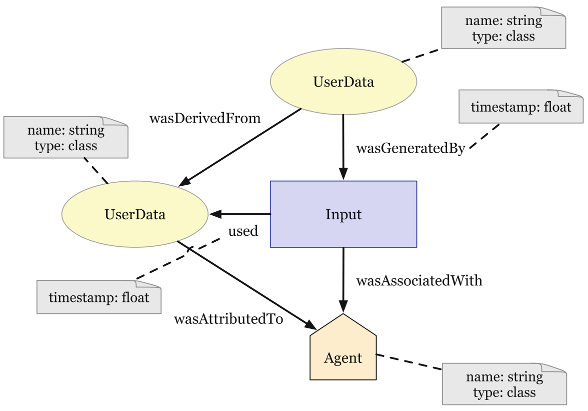

Fig. 2. PROV model for user input.

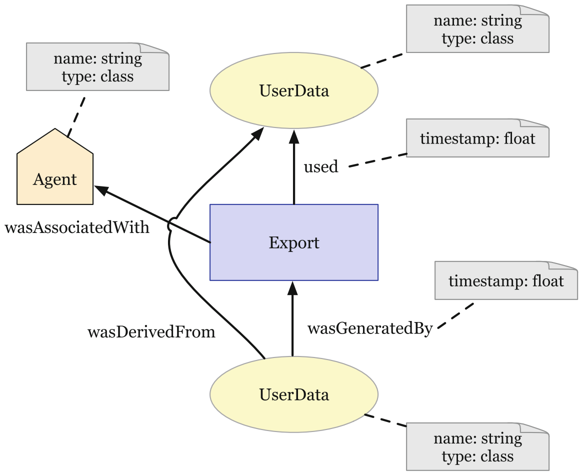

Fig. 3. PROV model for exporting data.

session on Mapping Data Access [19]), we developed a provenance model for QS workflows.

The possible activities in QS workflows are categorized into five abstract functionalities: *Input*, *Export*, *Request*, *Aggregation*, and *Visualization*. We defined a provenance sub model for each of these abstract functionalities.

Input. Specifies the functionality of using raw data from the user (through interaction or devices with sensors) and generating structured data from it (Fig. 2). Usually every QS workflow needs such a functionality to record data from the user.

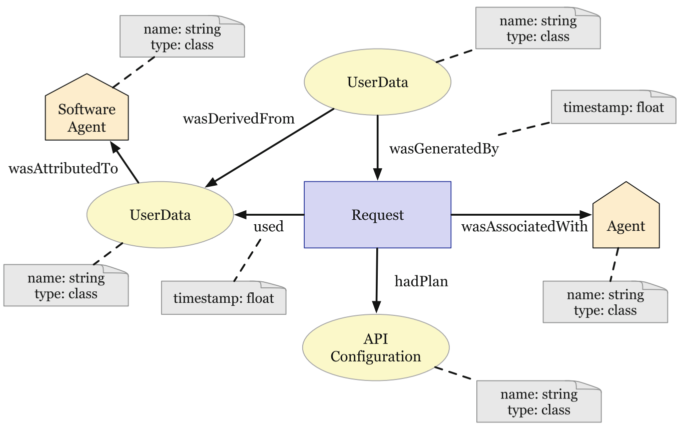

Fig. 4. PROV model for requesting data from a Web Service or Cloud service.

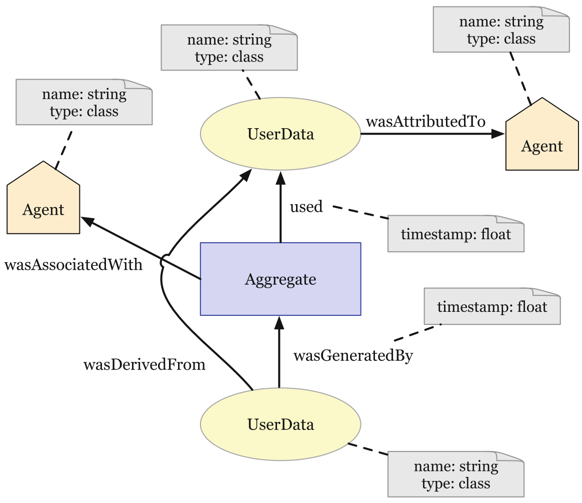

Fig. 5. PROV model for aggregating from two or more data sources.

Export. Specifies the functionality of exporting data from the current system into a format that is readable for other computer systems or humans (Fig. 3). Since a user usually uses more than one device, this functionality is processed multiple times, until the user see the visualized results.

Request. This functionality is in many QS workflows, where the user stores his data via Web Services or at Cloud Services. Often clients of these services requests data regularly from other Web Services or applications to update their data set (Fig. 4). Also, the user can trigger this functionality if the system does not update the data automatically.

Aggregation. A user usually records his data over time to interpret knowledge from it. This data needs to be aggregated with software capable of storing and updating data sets (Fig. 5). Many portals and desktop applications are aggregating heterogeneous data to create a deep knowledge about the user and help him understand his data [10].

Visualization. For many users who track themselves over a longer period, the visualization of their data over time in various graphics is a common approach to reveal knowledge of his data to them (Fig. 6).

4 Example: Visualizing Fitbit Data

To illustrate provenance recording, the following shows a very simple example. In this, steps data is requested from the Fitbit API [3], the data is cleaned and then visualized. We implemented the full example with Python and some supporting libraries, such as *python-fitbit* [11], *pandas* [13], and *matplotlib* [6]. In this example, we use a Python library [7] for storing the provenance and for generating a graph of it. During each step the actual Python command are preceded or followed by command for adding information to the provenance document.

Note: The following example is not complete. Some commands are left out. The full example is available online as Jupyter notebook[1].

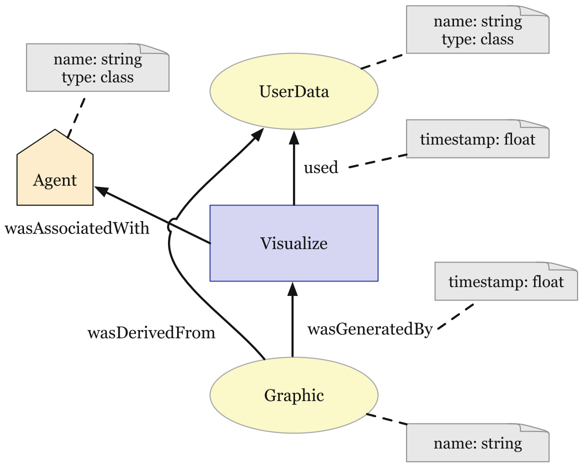

Fig. 6. PROV model for visualizing user data.

[1] Available at https://github.com/onyame/quantified_self_prov.

Loading Python Module and Initializing Empty PROV Document

```
from prov.model import ProvDocument
prov = ProvDocument()
```

Defining Namespaces

```
prov.add_namespace('qs', 'http://software.dlr.de/qs/')
prov.add_namespace('user', 'http://software.dlr.de/qs/user/')
prov.add_namespace('userdata', 'http://software.dlr.de/qs/userdata/')
prov.add_namespace('graphic', 'http://software.dlr.de/qs/graphic/')
prov.add_namespace('library', 'https://pypi.python.org/pypi/#')
prov.add_namespace('fitbit', 'https://www.fitbit.com/user/')
```

Adding Python Modules as PROV Entities

```
prov.entity('library:fitbit',
  {'prov:version': fitbit.__version__ })
prov.entity('library:pandas',
  {'prov:version': pd.__version__})
prov.entity('library:matplotlib',
  {'prov:version': matplotlib.__version__})
```

Adding User as PROV Agent

```
agent_user = prov.agent('user:onyame@googlemail.com',
  {'prov:type': 'prov:Person', 'qs:fitbit:id': '%s' % user_id})
```

Getting Values from Fitbit Web Service

```
t1 = strftime("%Y%m%dT%H%M%S%Z", gmtime())
ts = fitbit_client.time_series('activities/steps',
                               user_id=user_id, period='1y')
t2 = strftime("%Y%m%dT%H%M%S%Z", gmtime())
entity_timeseries = prov.entity('userdata:timeseries')
prov.activity('method:time_series', t1, t2,
  {'prov:type' : 'qs:request'})
prov.wasGeneratedBy(entity_timeseries, 'method:time_series', t2)
prov.used('method:time_series', 'library:fitbit')
prov.wasAttributedTo(entity_timeseries, agent_user)
```

Producing a Graph from the Time Series Data

```
steps.plot(label='Steps')
entity_plot_steps = prov.entity('graphic:steps')
activity_plot = prov.activity('method:matplotlib_plot', t1, t2,
  {'prov:type': 'qs:visualize'})
prov.used(activity_plot, entity_steps)
prov.wasGeneratedBy(entity_plot_steps, activity_plot, t2)
prov.wasDerivedFrom(entity_plot_steps, entity_steps)
prov.wasAssociatedWith(entity_plot_steps, agent_user)
```

The result is a provenance document represented as a graph (Fig. 7). This shows how the steps graphic (entity *graphic:steps*) is generated using data gathered from the Fitbit API via a library (entity *library:fitbit*).

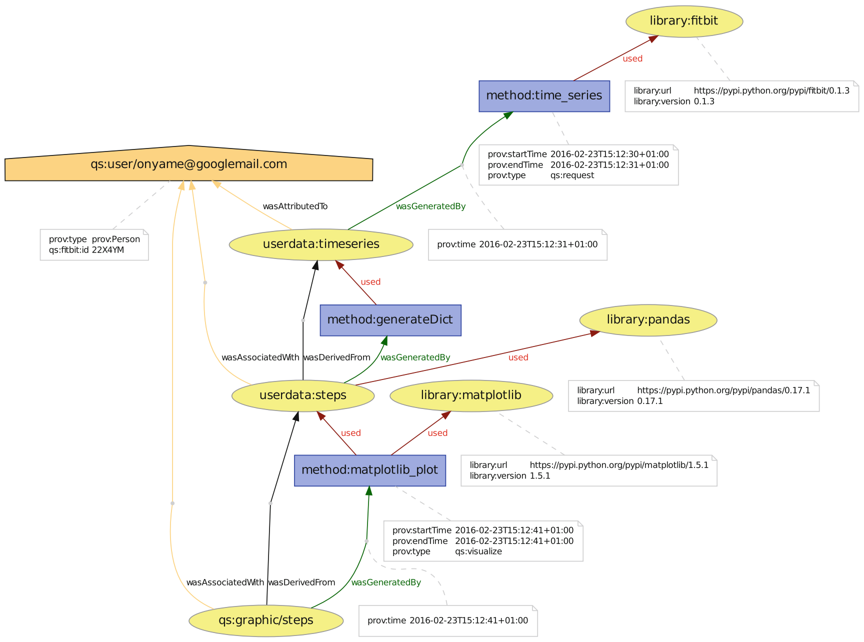

Fig. 7. Provenance of a graph with daily steps. The PROV document is available online in the ProvStore: https://provenance.ecs.soton.ac.uk/store/documents/113488/

5 Conclusions and Future Work

We provided an ontology and a provenance data model for Quantified Self data. Both are based on the open W3C standards PROV-O and PROV-DM. The provenance model reflects all essential data processing steps, which occur in todays QS applications. The provenance model in its current version is relatively abstract. This abstract definition allows modeling a great variety of QS data flows, but it should be improved to be more specific in upcoming versions.

Future work will focus on deployment of provenance recording in real-world QS applications. For example, the provenance of a whole workflow of steps data from a device or sensor, via smartphone apps and cloud services, to aggregation and visualization portals. The challenging task is to add provenance recording to legacy hardware and software in cases where manufacturers and companies are not collaborating.

Another future topic is to elaborate the queries of recorded provenance data. Based on provenance of full QS data flows meaningful insights should be generated. Important will be, to answer questions regarding security and safety of data.

Acknowledgements. We would like to thank Bojan Janisch (Bonn-Rhein-Sieg University of Applied Sciences), for his supportive work he did during his master project.

References

1. Bavoil, L., Callahan, S.P., Crossno, P.J., Freire, J., Vo, H.T.: VisTrails: enabling interactive multiple-view visualizations. In: pp. 135–142. IEEE (2005)
2. Choe, E.K., Lee, N.B., Lee, B., Pratt, W., Kientz, J.A.: Understanding quantified-selfers' practices in collecting and exploring personal data. In: Proceedings of the 32nd Annual ACM Conference on Human Factors in Computing Systems, pp. 1143–1152. ACM (2014)
3. Fitbit: Fitbit developer api (2016). https://dev.fitbit.com
4. Hoekstra, R., Groth, P.: PROV-O-Viz - understanding the role of activities in provenance. In: Ludäescher, B., Plale, B. (eds.) IPAW 2014. LNCS, vol. 8628, pp. 215–220. Springer, Heidelberg (2015)
5. Hoy, M.B.: Personal activity trackers and the quantified self. Med. Ref. Serv. Q. **35**(1), 94–100 (2016)
6. Hunter, J.D.: Matplotlib: a 2d graphics environment. Comput. Sci. Eng. **9**(3), 90–95 (2007)
7. Huynh, T.D.: A python library for W3C provenance data model supporting PROV-JSON import/export (2014). https://github.com/trungdong/prov
8. Huynh, T.D., Moreau, L.: ProvStore: a public provenance repository. In: Ludaescher, B., Plale, B. (eds.) IPAW 2014. LNCS, vol. 8628, pp. 275–277. Springer, Heidelberg (2015)
9. Janisch, B.: Developing an abstract Quantified Self Provenance-model. Master project, University of Applied Sciences Bonn-Rhein-Sieg (2015). http://elib.dlr.de/100752/
10. Jones, S.L.: Exploring correlational information in aggregated quantified self data dashboards. In: Adjunct Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2015 ACM International Symposium on Wearable Computers, pp. 1075–1080. ACM (2015)
11. Kelly, I.: python-fitbit - fitbit api python client implementation (2016). https://github.com/orcasgit/python-fitbit
12. Lebo, T., Sahoo, S., McGuinness, D., Belhajjame, K., Cheney, J., Corsar, D., Garijo, D., Soiland-Reyes, S., Zednik, S., Zhao, J.: PROV-O: The PROV ontology, 30 April 2013. http://www.w3.org/TR/2013/REC-prov-o-20130430/
13. Mckinney, W.: pandas: a foundational python library for data analysis and statistics. In: PyHPC 2011: Workshop on Python for High Performance and Scientific Computing, SC11, Seattle, WA, USA, 18 November 2011
14. Moreau, L., Groth, P., Cheney, J., Lebo, T., Miles, S.: The rationale of PROV. Web Seman. Sci. Serv. Agents World Wide Web **35**, Part 4, 235–257 (2015)
15. Moreau, L., Groth, P., Miles, S., Vazquez-Salceda, J., Ibbotson, J., Jiang, S., Munroe, S., Rana, O., Schreiber, A., Tan, V., Varga, L.: The provenance of electronic data. Commun. ACM **51**(4), 52–58 (2008)

16. Moreau, L., Missier, P., Belhajjame, K., B'Far, R., Cheney, J., Coppens, S., Cresswell, S., Gil, Y., Groth, P., Klyne, G., Lebo, T., McCusker, J., Miles, S., Myers, J., Sahoo, S., Tilmes, C.: PROV-DM: The PROV data model, 30 April 2013. http://www.w3.org/TR/2013/REC-prov-dm-20130430/
17. Noy, N.F., Mcguinness, D.L.: Ontology development 101: A guide to creating your first ontology (2001). http://www.ksl.stanford.edu/people/dlm/papers/ontology101/ontology101-noy-mcguinness.html
18. Picard, R., Wolf, G.: Sensor informatics and quantified self. IEEE J. Biomed. Health Inform. **19**(5), 1531 (2015)
19. QSEU14: Breakout: Mapping data access, 23 August 2014. https://forum.quantifiedself.com/t/breakout-mapping-data-access/995/4
20. Schreiber, A., Ney, M., Wendel, H.: The provenance store prOOst for the open provenance model. In: Groth, P., Frew, J. (eds.) IPAW 2012. LNCS, vol. 7525, pp. 240–242. Springer, Heidelberg (2012)