Quantified Self Comics

Andreas Schreiber

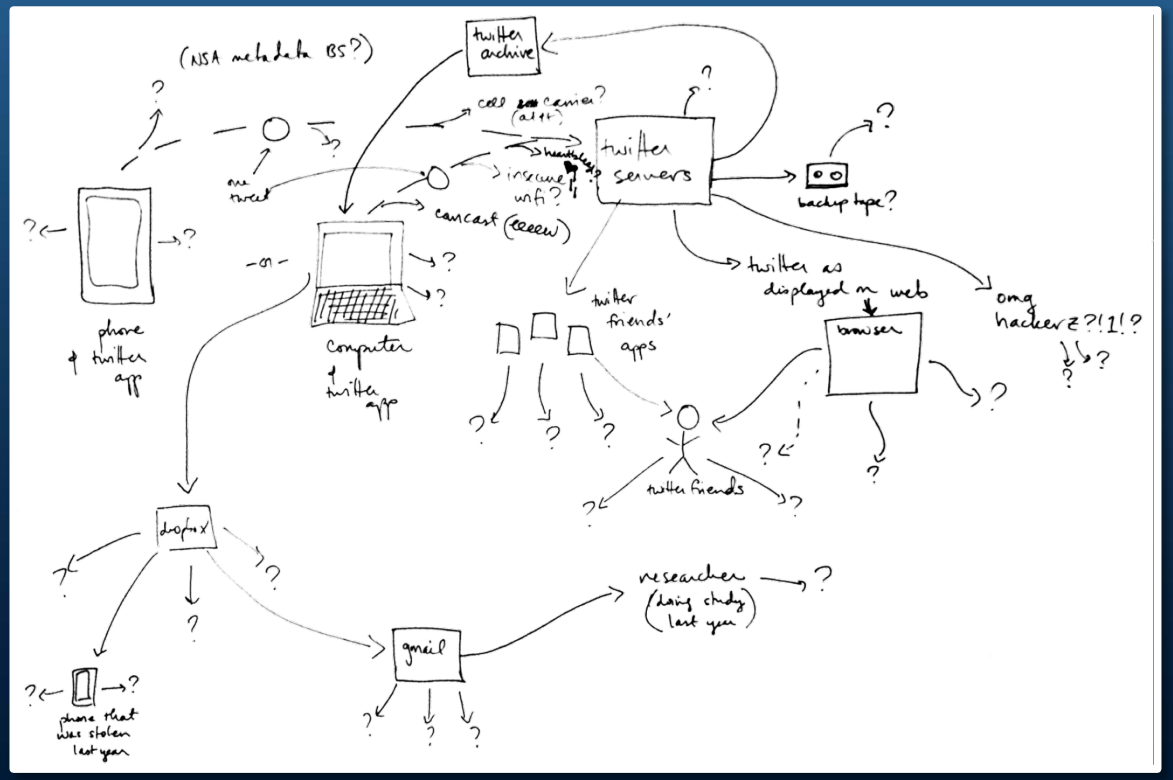
Introduction

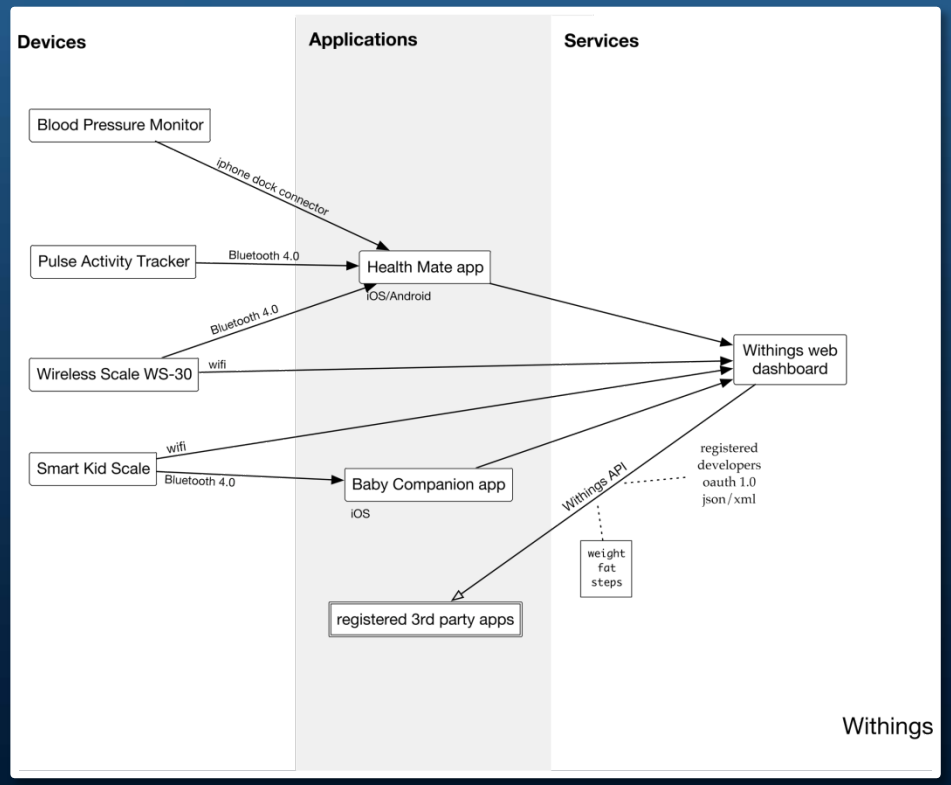
Deutsches Zentrum
für Luft- und Raumfahrt
German Aerospace Center

Foto: © Ann Christine Freuwörth, Wuppertal

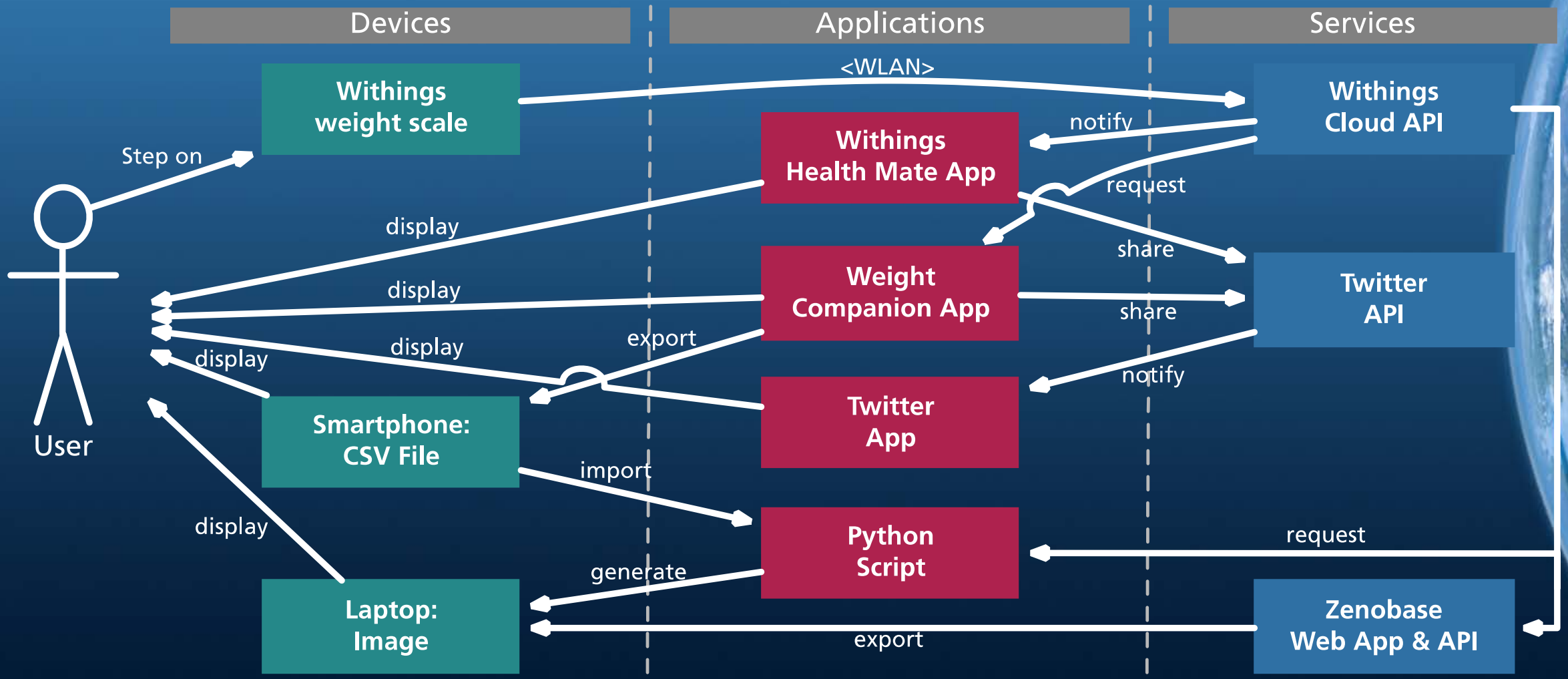
Understand, how QS data has been produced, processed, stored, accessed, …

Pictures from *Breakout Session on Mapping Data Access* (2014 QS Europe Conference, Amsterdam)
`https://forum.quantifiedself.com/t/breakout-mapping-data-access/995`





Example: My Weight Tracking Workflow



Questions related to QS Data and Activities

Data

- What data about the user were created during the activity X?
- What data about the user were automatically generated?
- What data about the user were derived from manual input?

Apps and Services

- Which activities support visualization of the users data?
- In which activities can the user input data?
- What processes are communicating data?

Access and Privacy

- What parties were involved in generating data X?
- What parties got access on data X?
- Can other parties see user’s data X?

Technology: Provenance of Data

Provenance is

information about entities, activities, and people involved in

producing a piece of data or thing,

which can be used to form

assessments about its quality, reliability or trustworthiness.

PROV W3C Working Group
https://www.w3.org/TR/prov-overview

Key Concepts of Provenance Model *PROV*

Entities

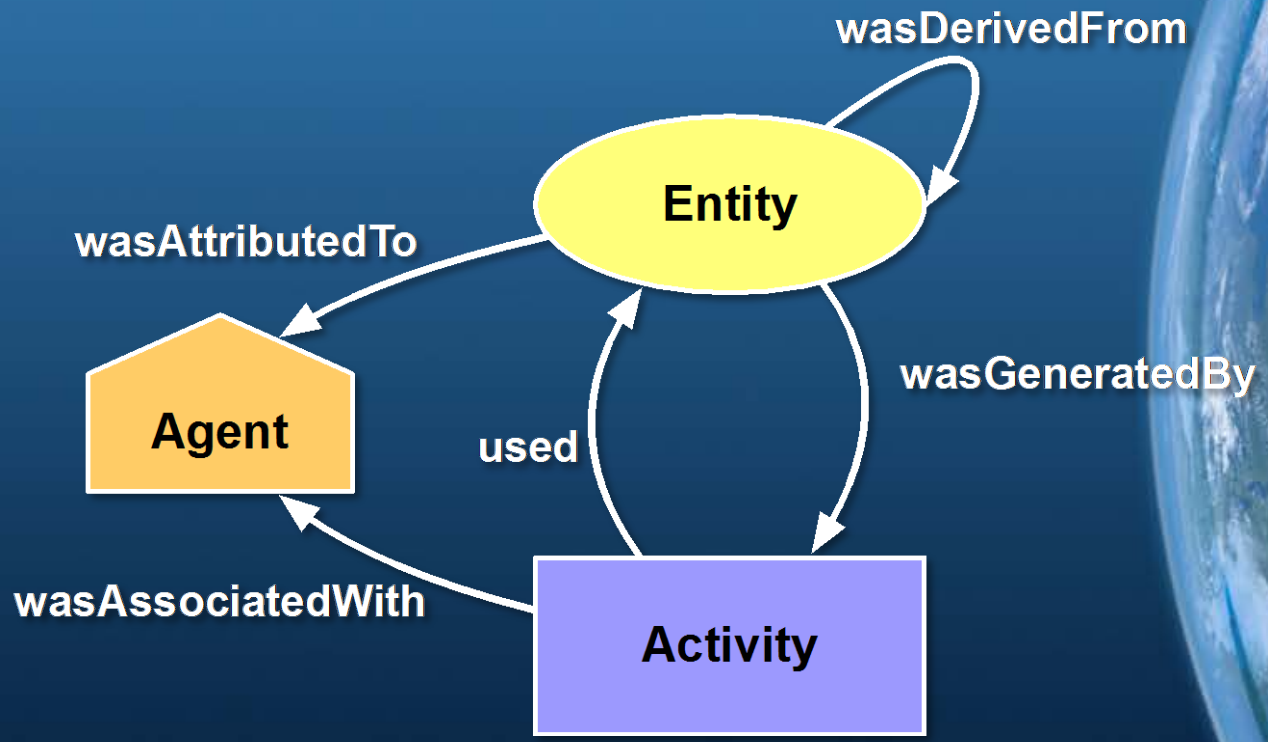
- Physical, digital, conceptual, or other kinds of things
- For example, documents, web sites, graphics, or data sets

Activities

- Activities *generate* new entities or make *use* of existing entities
- Activities could be actions or processes

Agents

- Agents takes a role in an activity and have the responsibility for the activity
- For example, persons, pieces of software, or organizations



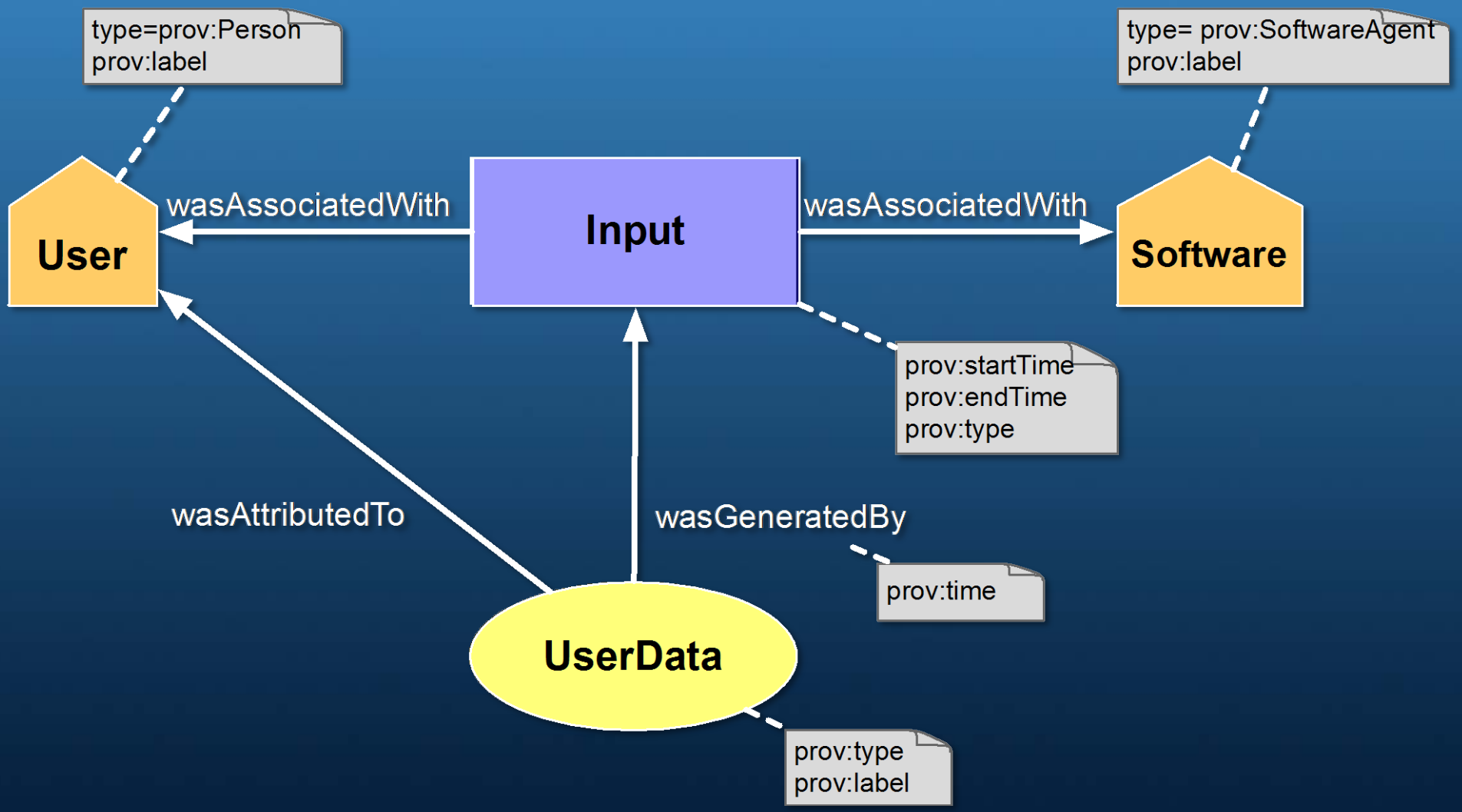
Provenance Model for Quantified Self

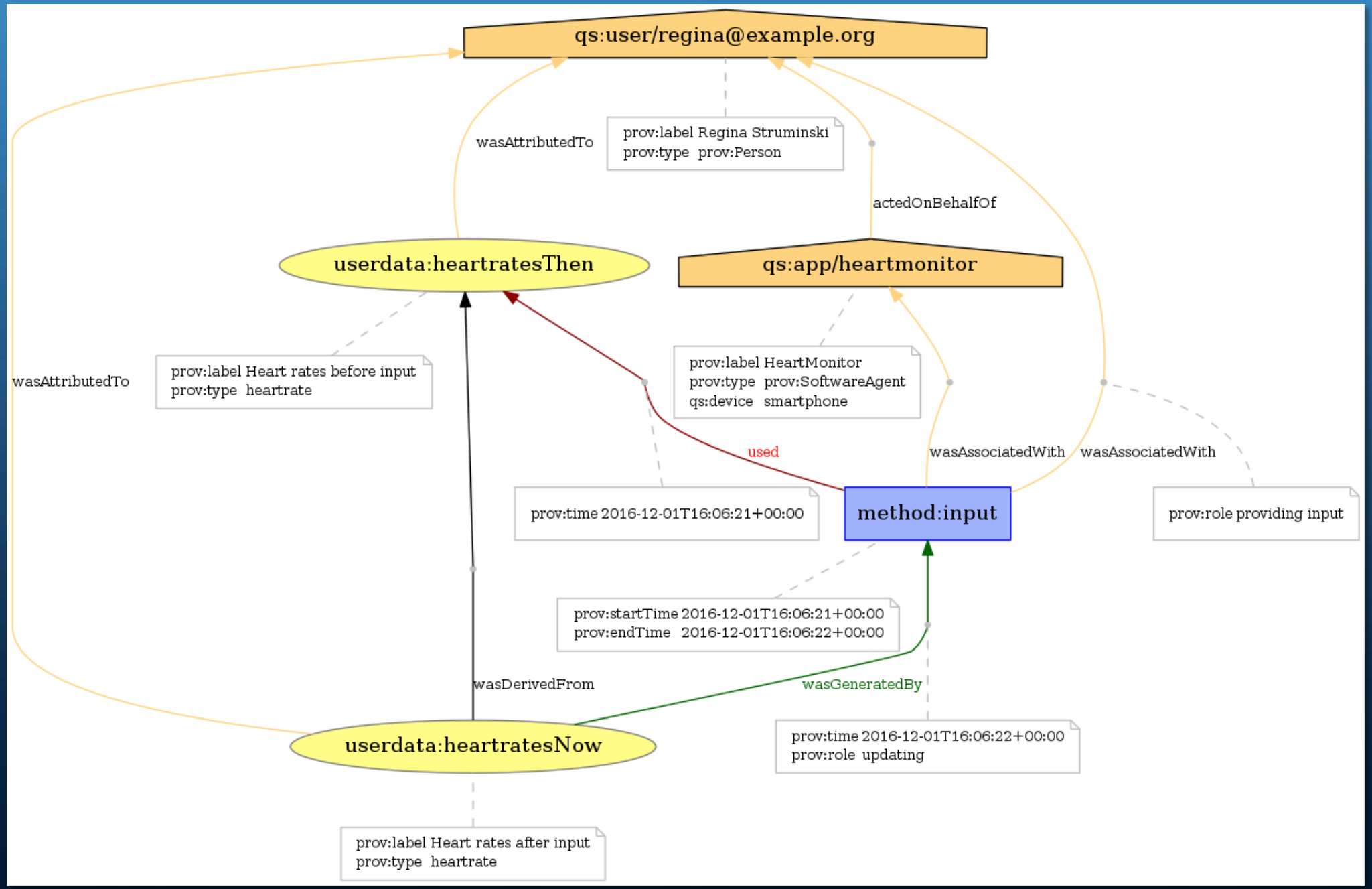
Sub models (templates) for basic *Activities*

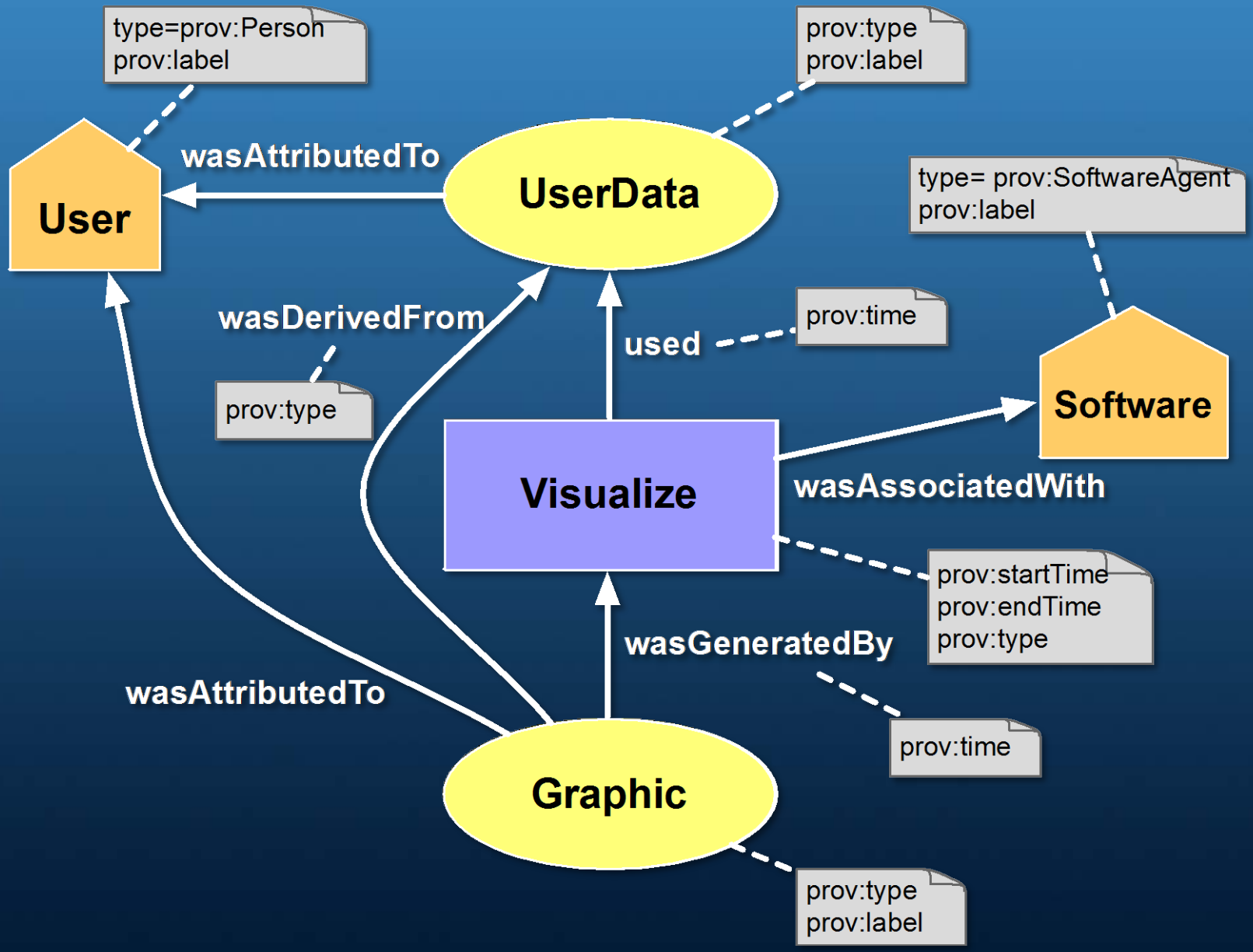
- Input
- Sensing
- Export
- Request
- Aggregate
- Visualize

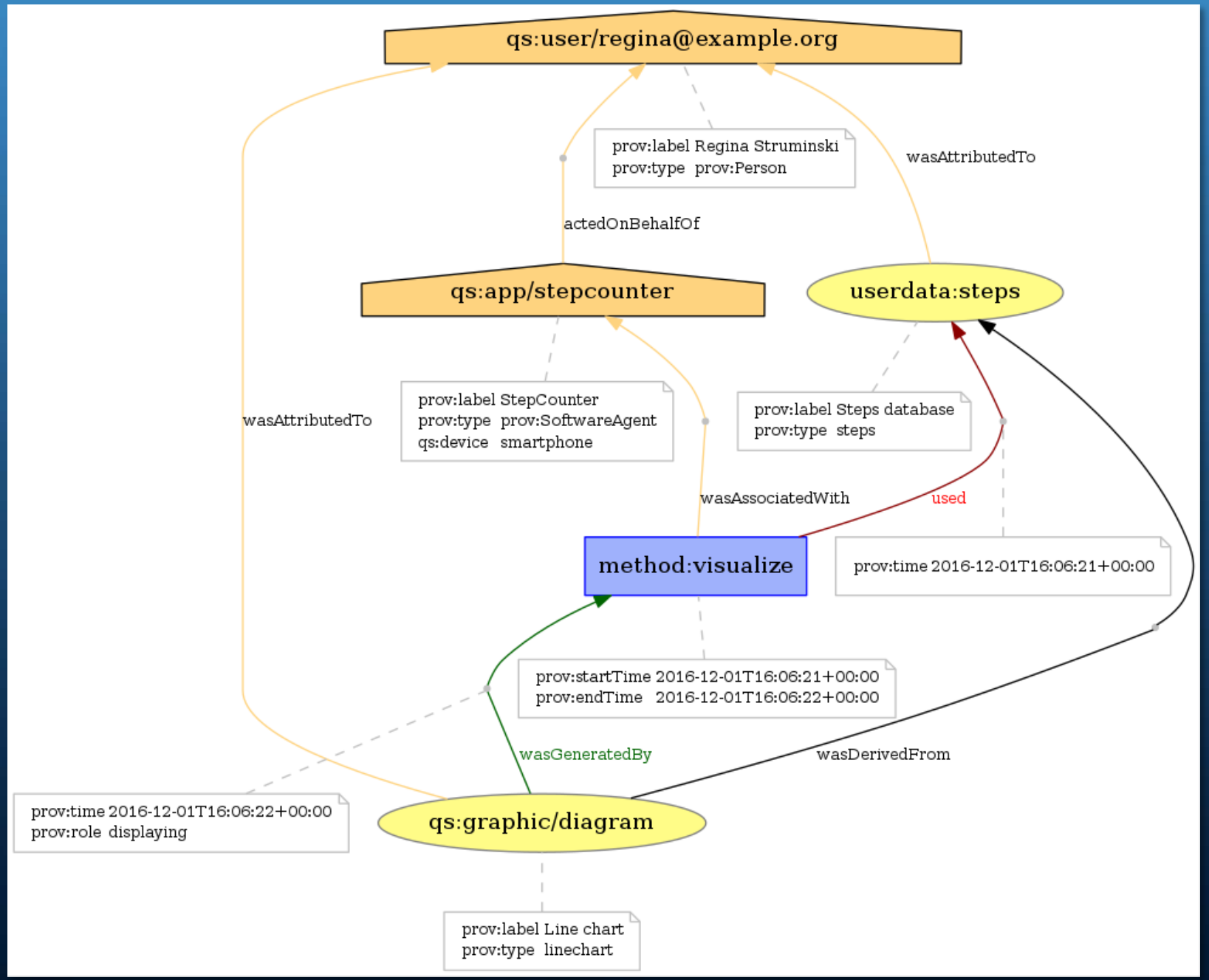
The activities generate or change data that is associated or attributed to *Agents*

- Users
- Software


type=prov:Person
prov:label
type= prov:SoftwareAgent
prov:label
User
wasAssociatedWith
Input
wasAssociatedWith
Software
prov:startTime
prov:endTime
prov:type
wasAttributedTo
wasGeneratedBy
prov:time
UserData
prov:type
prov:label


qs:user/regina@example.org
wasAttributedTo
prov:label Regina Struminski
prov:type prov:Person
actedOnBehalfOf
userdata:heartratesThen
qs:app/heartmonitor
wasAttributedTo
prov:label Heart rates before input
prov:type heartrate
prov:label HeartMonitor
prov:type prov:SoftwareAgent
qs:device smartphone
used
wasAssociatedWith
wasAssociatedWith
prov:time 2016-12-01T16:06:21+00:00
method:input
prov:role providing input
prov:startTime 2016-12-01T16:06:21+00:00
prov:endTime 2016-12-01T16:06:22+00:00
wasDerivedFrom
wasGeneratedBy
userdata:heartratesNow
prov:time 2016-12-01T16:06:22+00:00
prov:role updating
prov:label Heart rates after input
prov:type heartrate

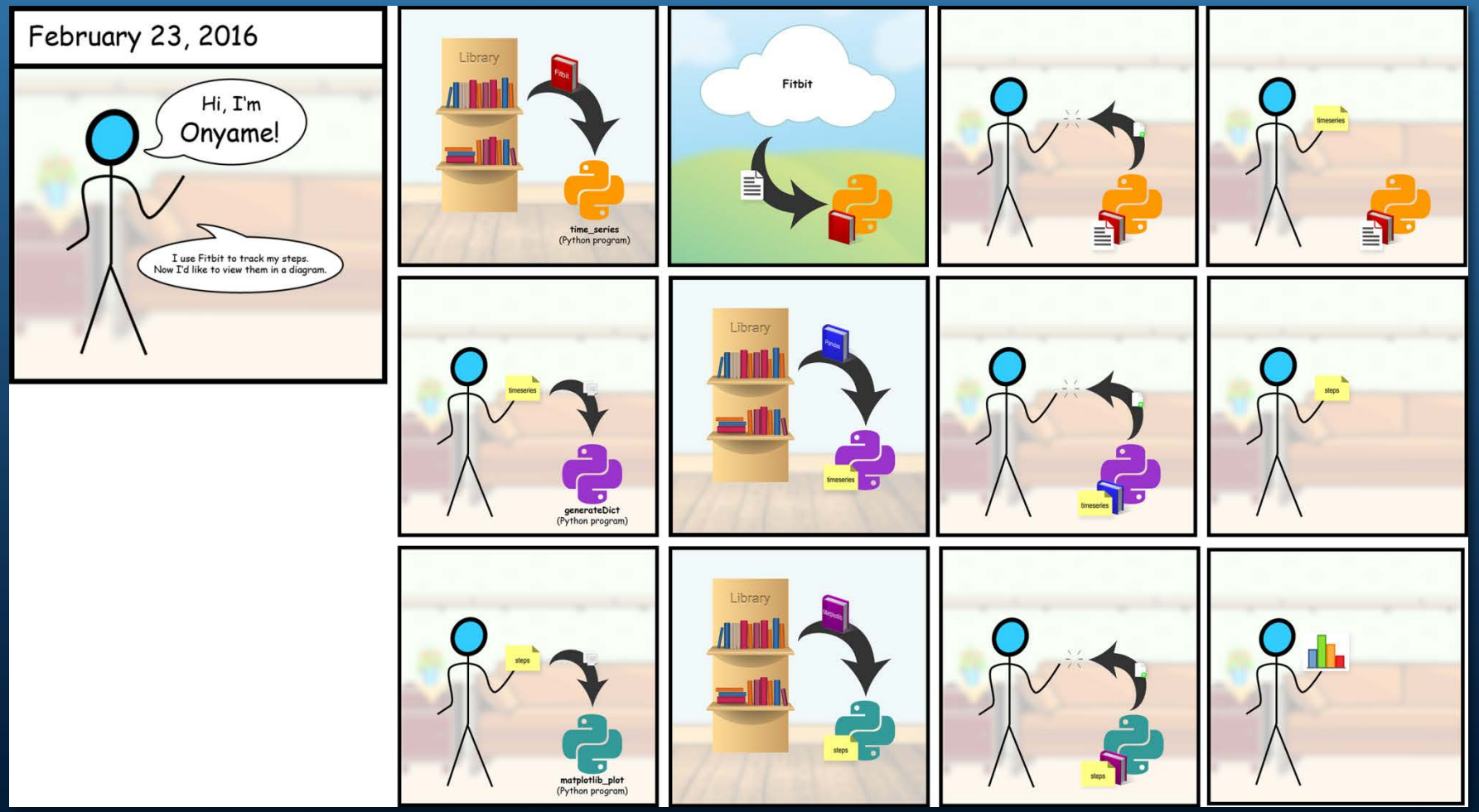

type=prov:Person
prov:label
prov:type
prov:label
wasAttributedTo
User
UserData
type= prov:SoftwareAgent
prov:label
wasDerivedFrom
prov:type
used
prov:time
Software
Visualize
wasAssociatedWith
prov:startTime
prov:endTime
prov:type
wasGeneratedBy
wasAttributedTo
prov:time
Graphic
prov:type
prov:label


qs:user/regina@example.org
prov:label Regina Struminski
prov:type prov:Person
wasAttributedTo
actedOnBehalfOf
qs:app/stepcounter
userdata:steps
wasAttributedTo
prov:label StepCounter
prov:type prov:SoftwareAgent
qs:device smartphone
prov:label Steps database
prov:type steps
wasAssociatedWith
used
method:visualize
prov:time 2016-12-01T16:06:21+00:00
prov:startTime 2016-12-01T16:06:21+00:00
prov:endTime 2016-12-01T16:06:22+00:00
wasGeneratedBy
wasDerivedFrom
prov:time 2016-12-01T16:06:22+00:00
prov:role displaying
qs:graphic/diagram
prov:label Line chart
prov:type linechart

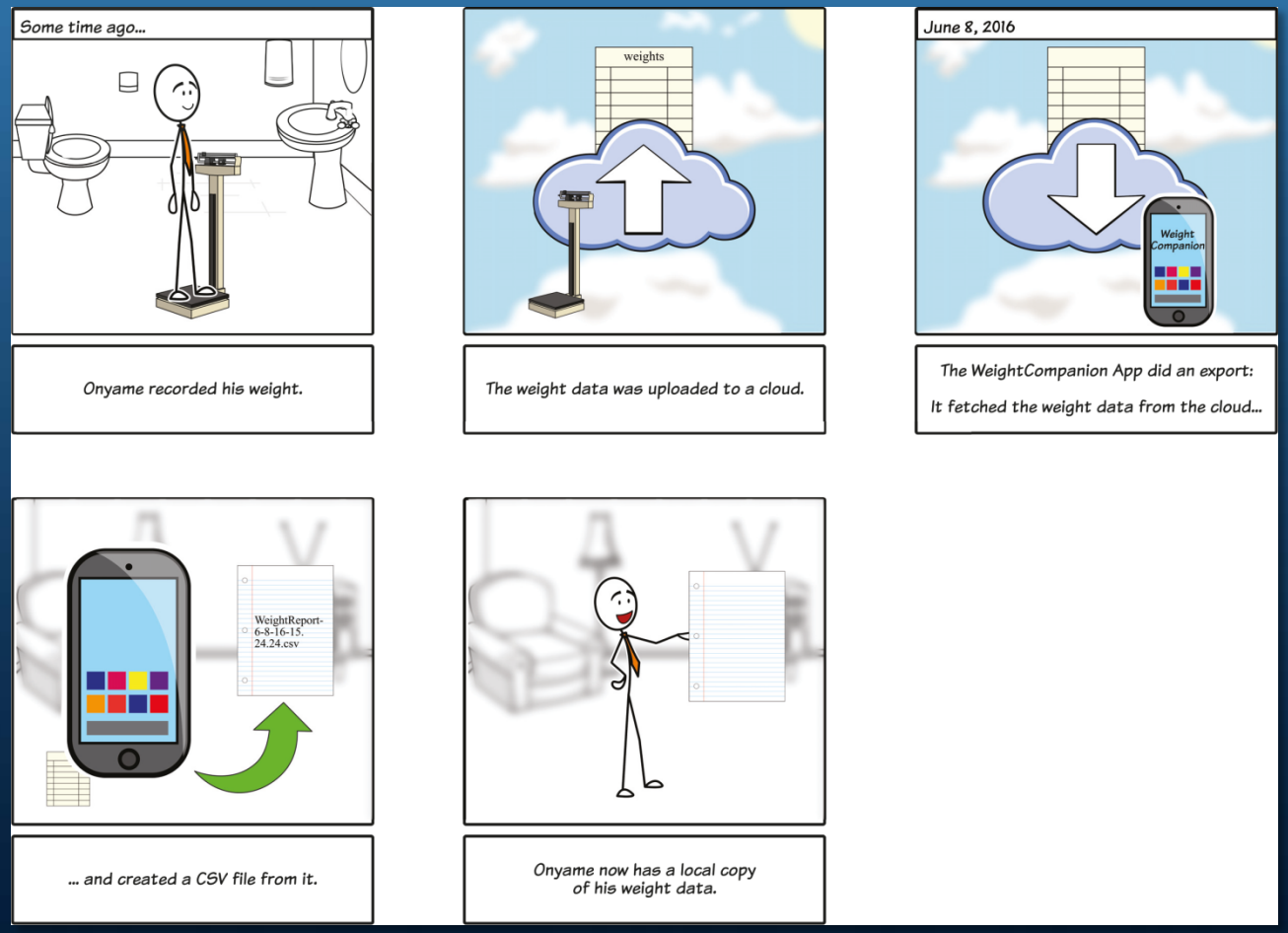
```
document
  prefix userdata <http://software.dlr.de/qs/userdata/>
  prefix qs <http://software.dlr.de/qs/>
  prefix graphic <http://software.dlr.de/qs/graphic/>
  prefix app <http://software.dlr.de/qs/app/>
  prefix user <http://software.dlr.de/qs/user/>
  prefix device <http://software.dlr.de/qs/device/>
  prefix method <http://www.java.com>

  wasGeneratedBy(qs:graphic/diagram, method:visualize, 2016-12-01T16:06:22+00:00, [prov:role="displaying"])
  activity(method:visualize, 2016-12-01T16:06:21+00:00, 2016-12-01T16:06:22+00:00)
  entity(qs:graphic/diagram, [prov:type="linechart", prov:label="Line chart"])
  entity(userdata:steps, [prov:type="steps", prov:label="Steps database"])
  agent(qs:user/regina@example.org, [prov:type="prov:Person", prov:label="Regina Struminski"])
  agent(qs:app/stepcounter, [prov:type="prov:SoftwareAgent", qs:device="smartphone", prov:label="StepCounter"])
  wasAttributedTo(qs:graphic/diagram, qs:user/regina@example.org)
  wasAttributedTo(userdata:steps, qs:user/regina@example.org)
  actedOnBehalfOf(qs:app/stepcounter, qs:user/regina@example.org, -)
  used(method:visualize, userdata:steps, 2016-12-01T16:06:21+00:00)
  wasDerivedFrom(qs:graphic/diagram, userdata:steps, -, -, -)
  wasAssociatedWith(method:visualize, qs:app/stepcounter, -)
endDocument
```

Standard Graph Visualizations and Textual Representations of Provenance Data are not Easy to Understand by Non-experts

Idea: Provenance Visualization Using Comics



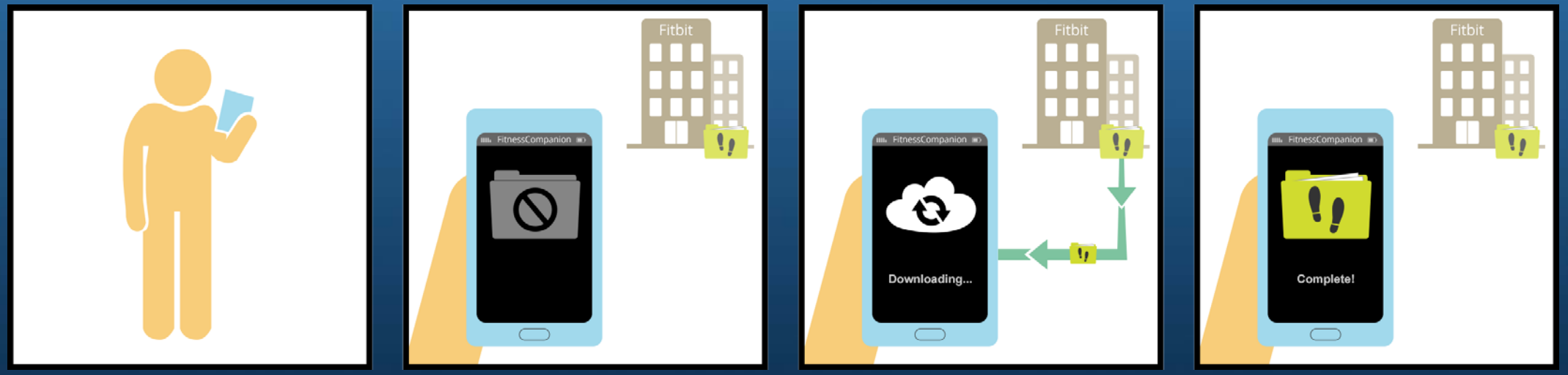
First Sketches



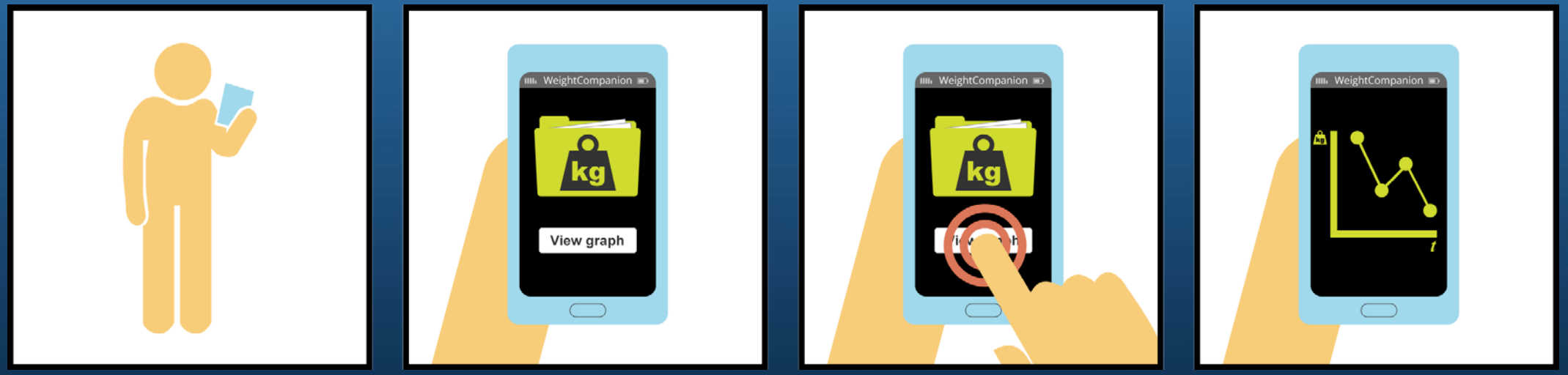
Current Style

Single Comic Strip Shows a Single Data-related Action

Communicate to People Where Data is Stored

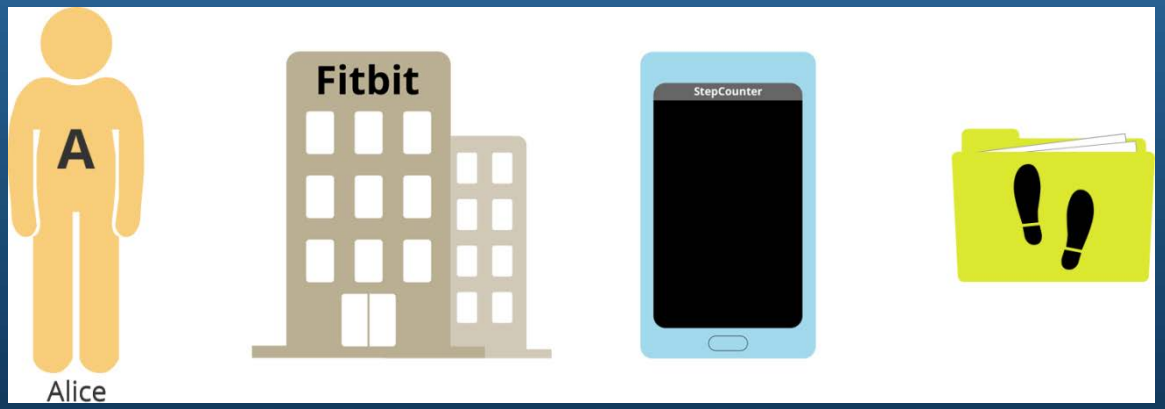


Understand How Data is Analyzed



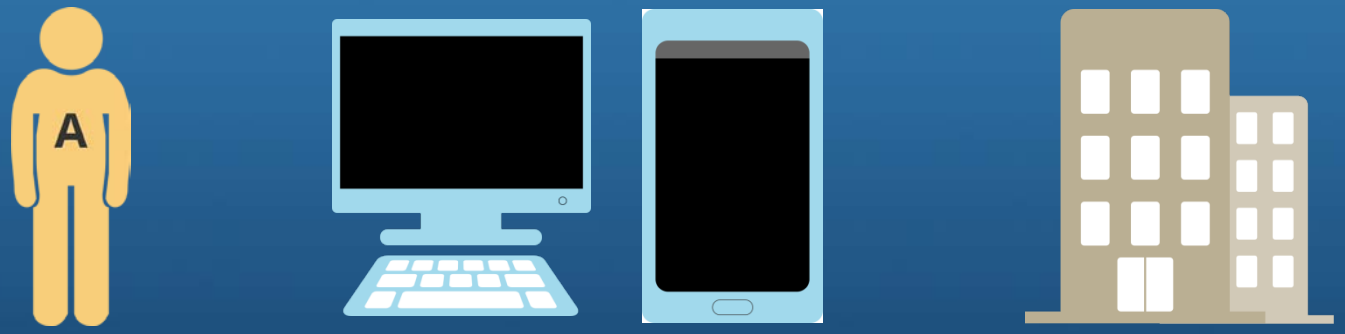
Distinctive Features

- Shapes
- Colors
- Icons
- Letters
- Labels

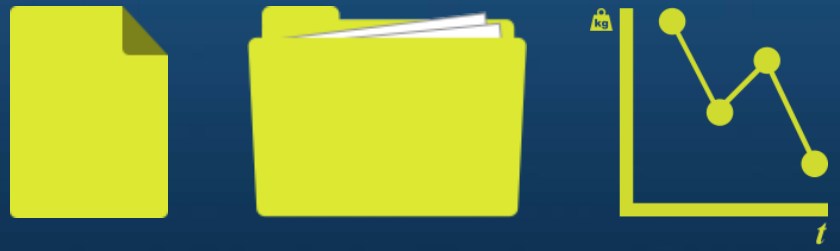


Representation of Provenance Elements

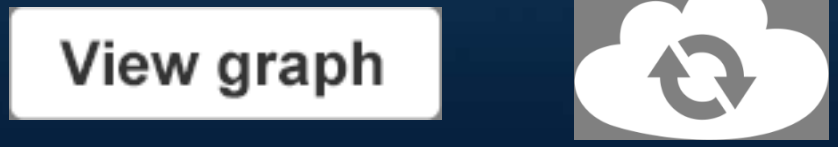
Agents



Entities



Activity-related



User Study

Sheet 1

Please imagine the following scenario:

You wear a **fitness bracelet** every day,
counting the steps you take.

On your smartphone, there is an app called **„StepsCompanion“**,
which automatically syncs with your bracelet.

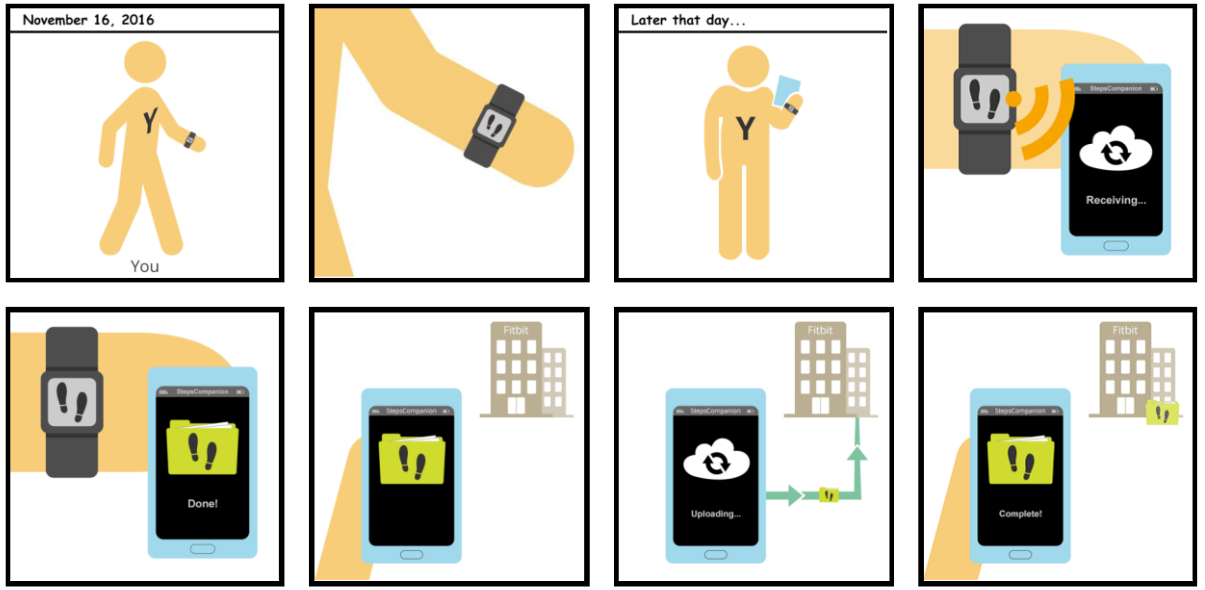
This way you can always view on your phone
how much you have already walked during the day.

You are now viewing your steps from November 16, 2016 in the app:

The button *"Origin of this data"* makes you curious.

So you tap it and get to see the following:

Test subject _____ Page 1/2



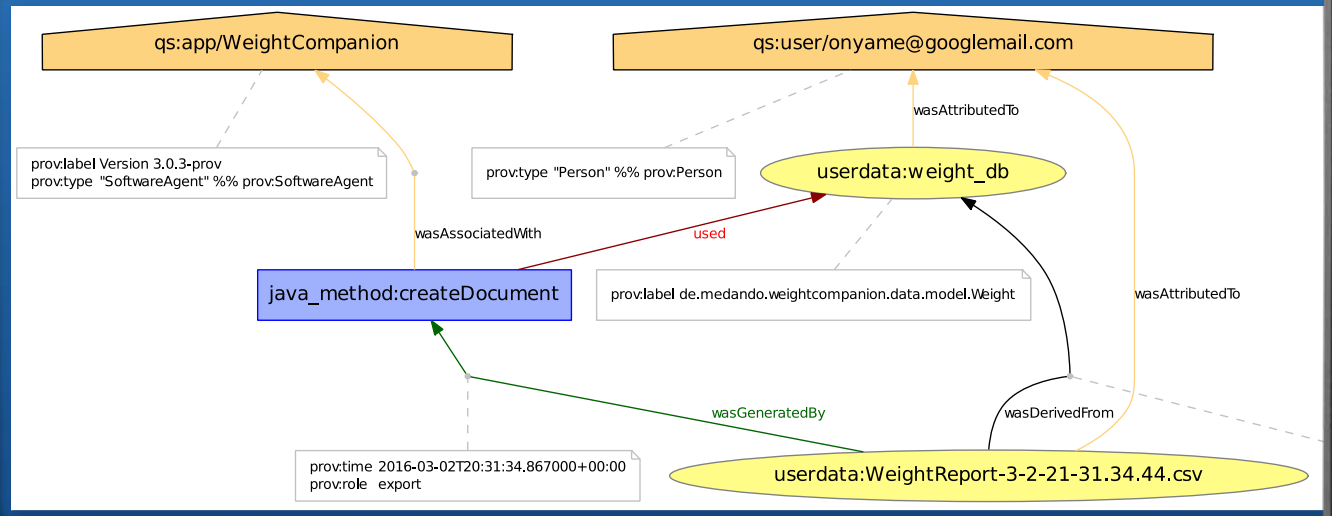
Please take your time to inspect and interpret these pictures.

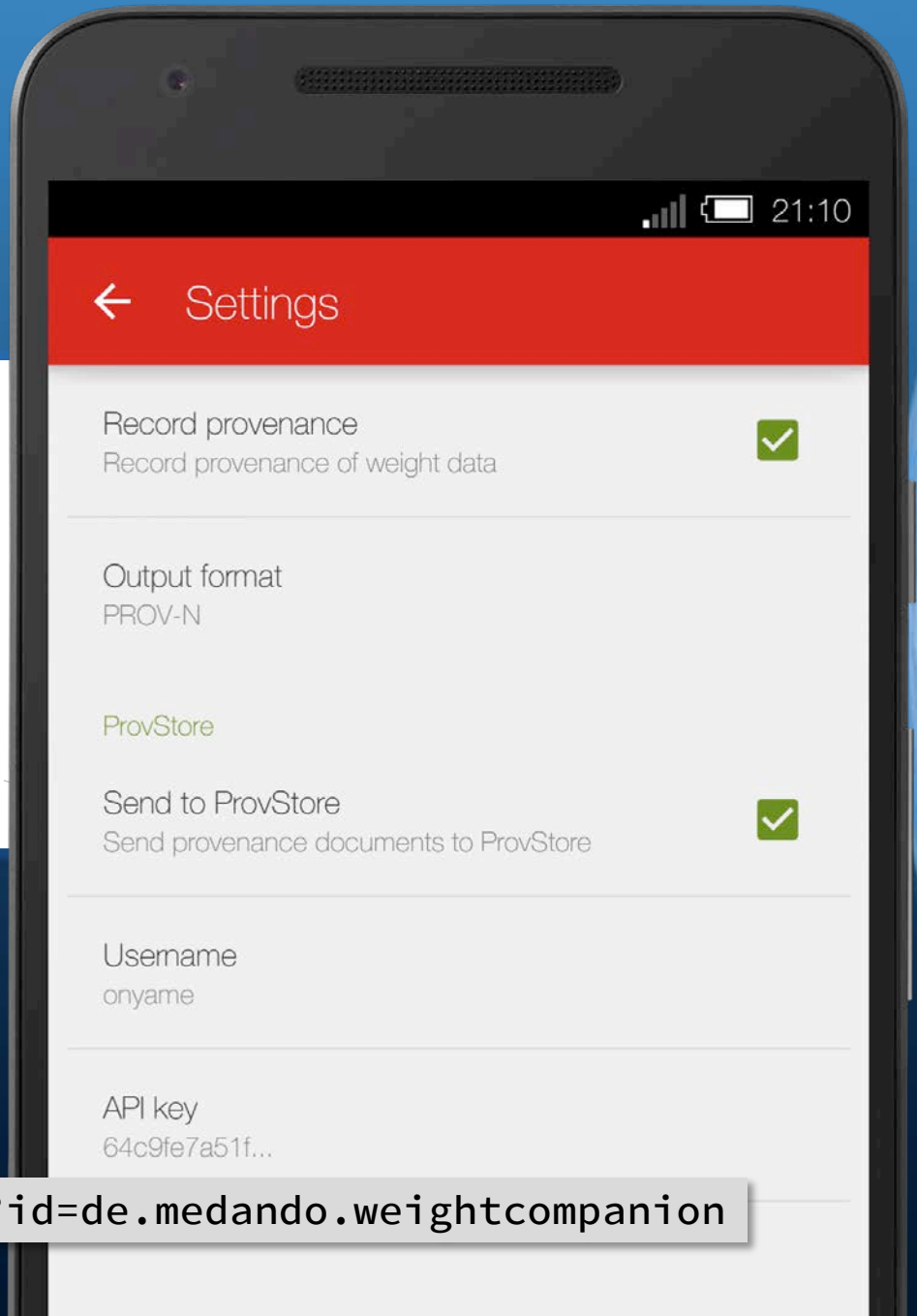
When you are ready, please tell the examiner what the pictures convey in your understanding.

Feel free to elaborate in great detail – mention anything that occurs to you or catches your attention.

Test subject _____ Page 2/2

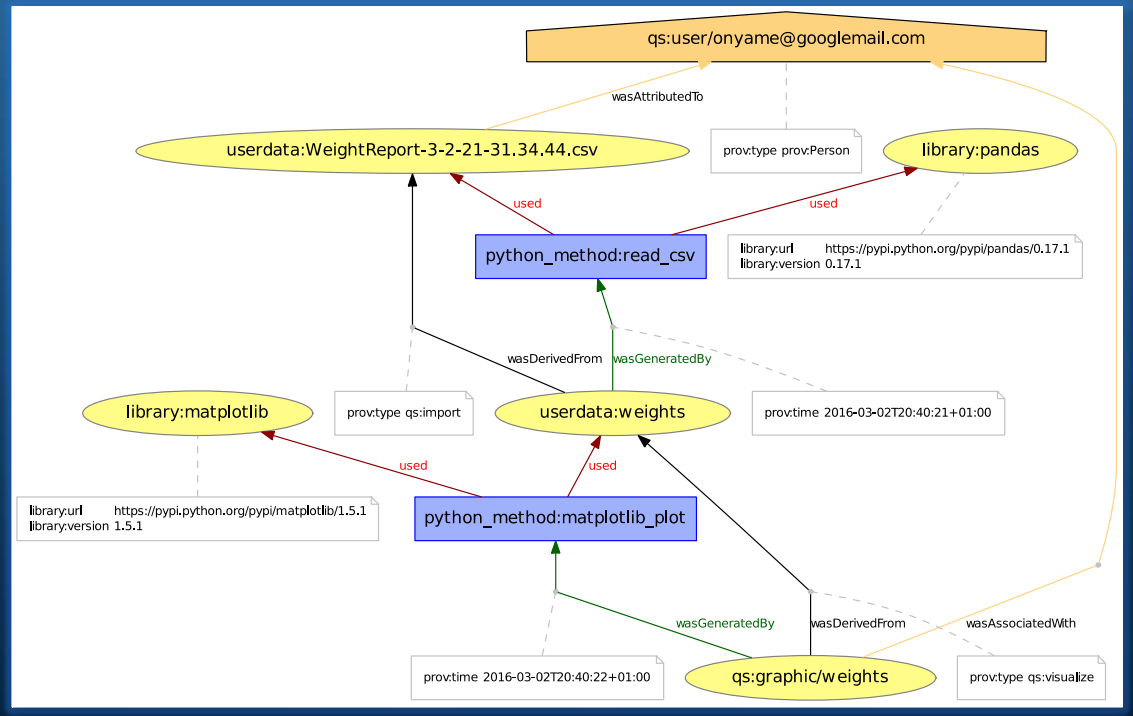
Collecting QS Provenance
Weight Tracking App



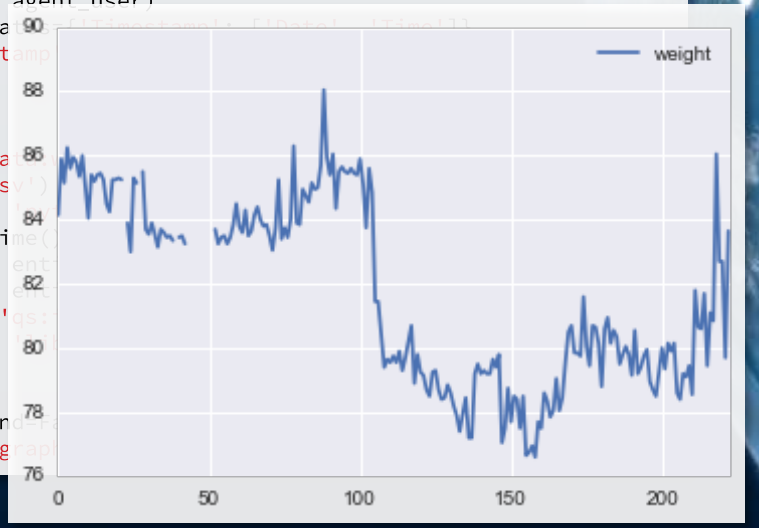


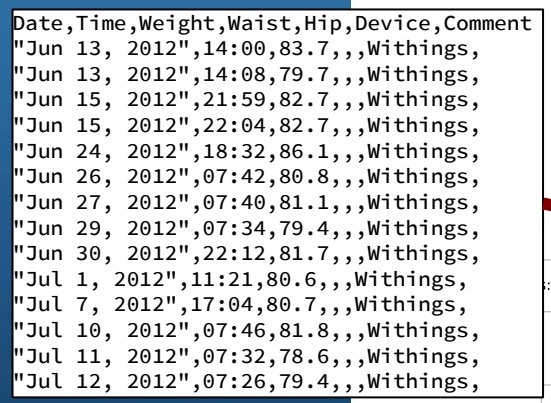
https://play.google.com/store/apps/details?id=de.medando.weightcompanion

Collecting QS Provenance
Visualization with Python Script



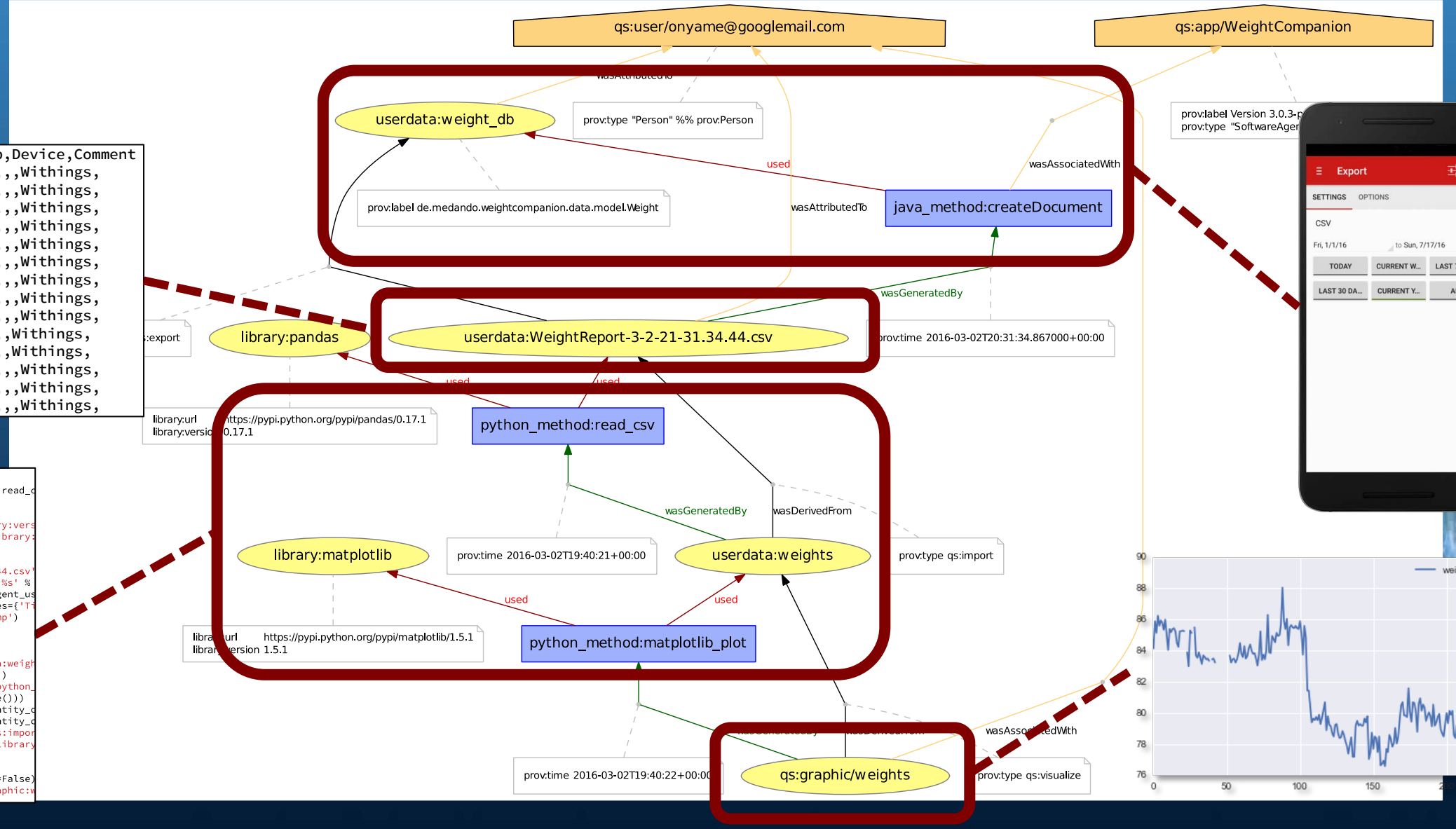

```python
# Provenance-related Imports
from prov.model import ProvDocument, PROV
from provstore.api import Api
from time import gmtime, strftime

# Create a new provenance document
prov = ProvDocument()

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

# The user
agent_user = prov.agent('user:onyame@googlemail.com', {'prov:type': PROV['Person']})

# Application Import
from pandas import DataFrame, Series, read_csv
import matplotlib.pyplot as plt

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

# Import weights from CSV file
WC_FILE = 'WeightReport-3-2-21-31.34.44.csv'
entity_csvfile = prov.entity('userdata:%s' % WC_FILE)
prov.wasAttributedTo(entity_csvfile, agent_user)
wc_data = read_csv(WC_FILE, parse_da
                   index_col='Timest

# Get just the weights
weights = wc_data[['Weight']]
entity_weights = prov.entity('userda
prov.activity('python_method:read_cs
prov.wasGeneratedBy(entity_weights,
    strftime('%Y%m%dT%H%M%S%Z', gmti
prov.used('python_method:read_csv',
prov.wasDerivedFrom(entity_weights,
    other_attributes={'prov:type': '
prov.used('python_method:read_csv',

# Plot the weights
weights.plot(title = 'Weight', legen
entity_plot_weights = prov.entity('g
```

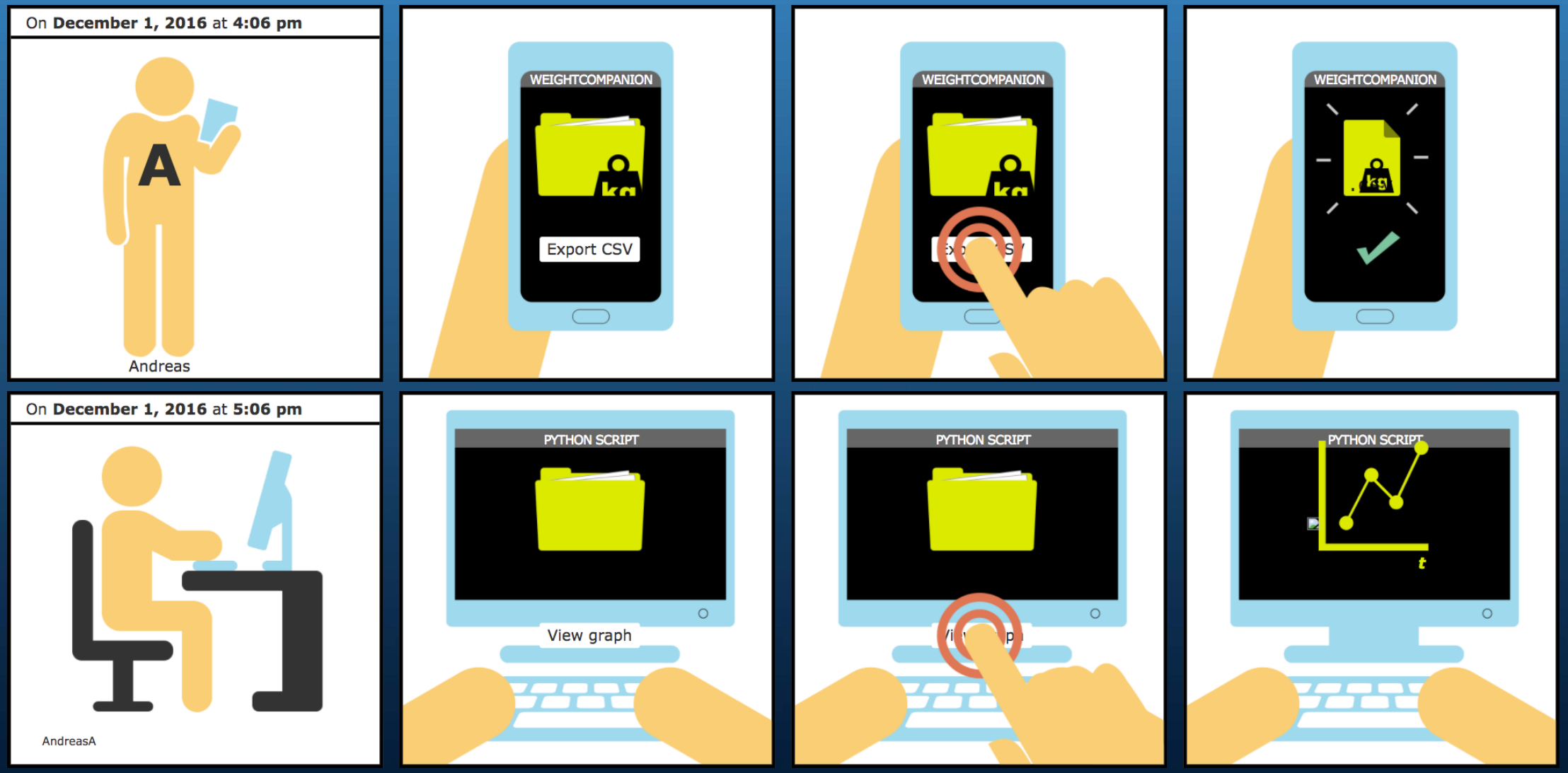




```
Date,Time,Weight,Waist,Hip,Device,Comment
"Jun 13, 2012",14:00,83.7,,,Withings,
"Jun 13, 2012",14:08,79.7,,,Withings,
"Jun 15, 2012",21:59,82.7,,,Withings,
"Jun 15, 2012",22:04,82.7,,,Withings,
"Jun 24, 2012",18:32,86.1,,,Withings,
"Jun 26, 2012",07:42,80.8,,,Withings,
"Jun 27, 2012",07:40,81.1,,,Withings,
"Jun 29, 2012",07:34,79.4,,,Withings,
"Jun 30, 2012",22:12,81.7,,,Withings,
"Jul 1, 2012",11:21,80.6,,,Withings,
"Jul 7, 2012",17:04,80.7,,,Withings,
"Jul 10, 2012",07:46,81.8,,,Withings,
"Jul 11, 2012",07:32,78.6,,,Withings,
"Jul 12, 2012",07:26,79.4,,,Withings,
```

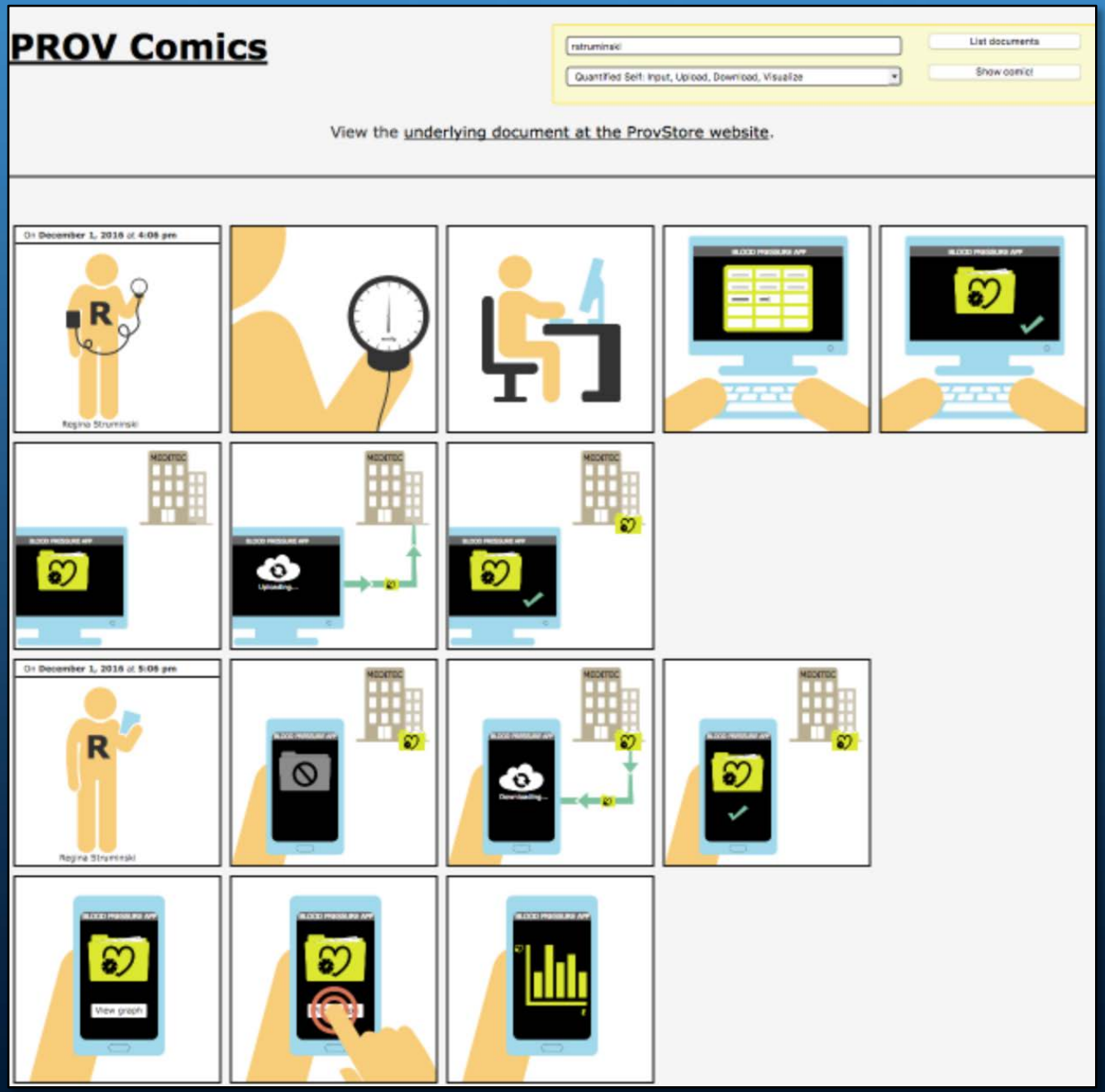
```
# Application Import
from pandas import DataFrame, Series, read_c
import matplotlib.pyplot as plt

prov.entity('library:pandas', {'library:vers
prov.entity('library:matplotlib', {'library:

# Import weights from CSV file
WC_FILE = 'WeightReport-3-2-21-31.34.44.csv'
entity_csvfile = prov.entity('userdata:%s' %
prov.wasAttributedTo(entity_csvfile, agent_us
wc_data = read_csv(WC_FILE, parse_dates={'Ti
                   index_col='Timestamp')

# Get just the weights
weights = wc_data[['Weight']]
entity_weights = prov.entity('userdata:weigh
prov.activity('python_method:read_csv')
prov.wasGeneratedBy(entity_weights, 'python_
    strftime('%Y%m%dT%H%M%S%Z', gmtime()))
prov.used('python_method:read_csv', entity_c
prov.wasDerivedFrom(entity_weights, entity_c
    other_attributes={'prov:type': 'qs:impor
prov.used('python_method:read_csv', 'library

# Plot the weights
weights.plot(title = 'Weight', legend=False)
entity_plot_weights = prov.entity('graphic:w
```




On December 1, 2016 at 4:06 pm
A
Andreas
WEIGHTCOMPANION
Export CSV
WEIGHTCOMPANION
WEIGHTCOMPANION
On December 1, 2016 at 5:06 pm
AndreasA
PYTHON SCRIPT
View graph
PYTHON SCRIPT
PYTHON SCRIPT
t

PROV Comics
Web Applications

http://provcomics.de



Current Status and Future Work

References

- Schreiber A., Seider D. (2016) ***Towards Provenance Capturing of Quantified Self Data***. In: Provenance and Annotation of Data and Processes. IPAW 2016. Lecture Notes in Computer Science, vol 9672. Springer, Cham

- Schreiber A., Struminski R. (2017) ***Tracing Personal Data Using Comics***. In: Universal Access in Human-Computer Interaction. Methods, Techniques, and Best Practices: 11th Internatioal Conference. HCI International, Vancouver, Canada

Future Work

- Different comic styles
- Quantitative comics
  - Geographical information
  - Glyph-based depiction
- … and other technical improvements

Thank You!

Andreas Schreiber

www.andreas-schreiber.net

andreas.schreiber@dlr.de


@onyame