Data Provenance

NDBI046: Practical class 9

Prerequisite: Setting up Python (Linux, macOS)

- *Check* which *version of Python* is installed (if any)
  - If Python 3 is not installed, download the (latest) version of Python[#1] and follow the installation
- Once installed, *create* any *folder for your NDBI046 project* and navigate to it, e.g., ~/Projects/python-ndbi046
- *Create* your *Python environment*, e.g., ndbi046_env
- *Activate* you Python environment
- *Install* the required *packages*
  - Download the requirements.txt file from the practical class website
  - You may also *install additional packages*
  - You may always export the list of installed packages to a file
- Exit the Python environment after completing the practical class (not before)
  - You can return to the environment at any time by activating it

```
1  % python3 --version
2
3  % mkdir ~/Projects/python-ndbi046
4  % cd ~/Projects/python-ndbi046
5
6  % python3 -m venv ndbi046_env
7
8  % source ndbi046_env/bin/activate
9
10 (ndbi046_env) % pip install -r requirements.txt
11
12 (ndbi046_env) % pip install pandas
13
14 (ndbi046_env) % pip freeze > requirements.txt
15
16 (ndbi046_env) % deactivate
17
18 % cat requirements.txt
```

checking the installed version

creating and activating a virtual environment

installation of the required packages

export of installed packages

list installed packages

deactivating a virtual environment

#1 https://www.python.org/downloads/

Prerequisite: Setting up Python (Windows)

- *Check* which *version of Python* is installed (if any)
  - If Python 3 is not installed, download the (latest) version of Python[#1] and follow the installation
- Once installed, *create* any *folder for your NDBI046 project* and navigate to it, e.g., C:\Projects\python-ndbi046
- *Create* your *Python environment*, e.g., ndbi046_env
- *Activate* you Python environment
- *Install* the required *packages*
  - Download the requirements.txt file from the practical class website
  - You may also *install additional packages*
  - You may always export the list of installed packages to a file
- Exit the Python environment after completing the practical class (not before)
  - You can return to the environment at any time by activating it

checking the installed version

```
> python3 --version

> mkdir C:\Projects\python-ndbi046
> cd C:\Projects\python-ndbi046

> python3 -m venv ndbi046_env

> ndbi046_env\Scripts\activate.bat

(ndbi046_env) > pip install -r requirements.txt

(ndbi046_env) > pip install pandas

(ndbi046_env) > pip freeze > requirements.txt

(ndbi046_env) > deactivate

> type requirements.txt
```

creating and activating a virtual environment

installation of the required packages

export of installed packages

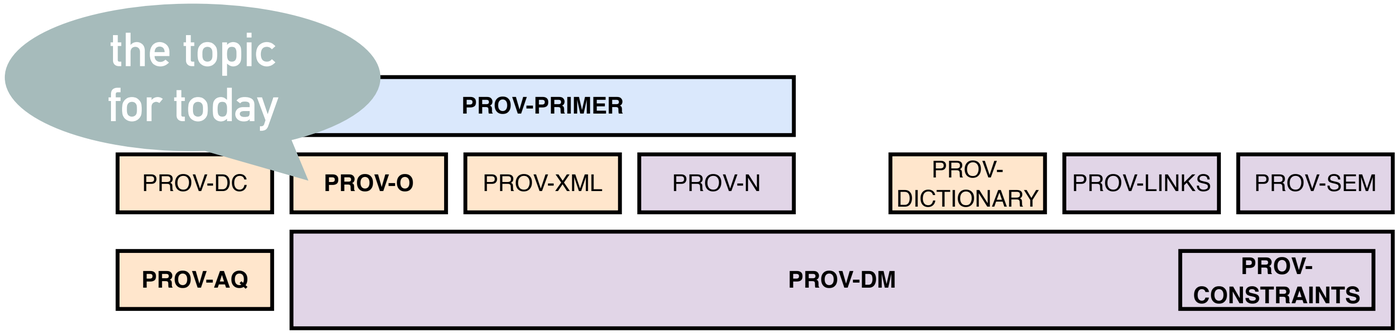
list installed packages

deactivating a virtual environment

#1 https://www.python.org/downloads/

Data provenance

- Provenance includes *details of the actors*, *activities*, *and contributors* in the production of the data
  - Helps to *assess* the *data reliability and quality*
- The PROV family enables interoperable provenance exchange in heterogeneous environments such as the Web through defined models and serializations

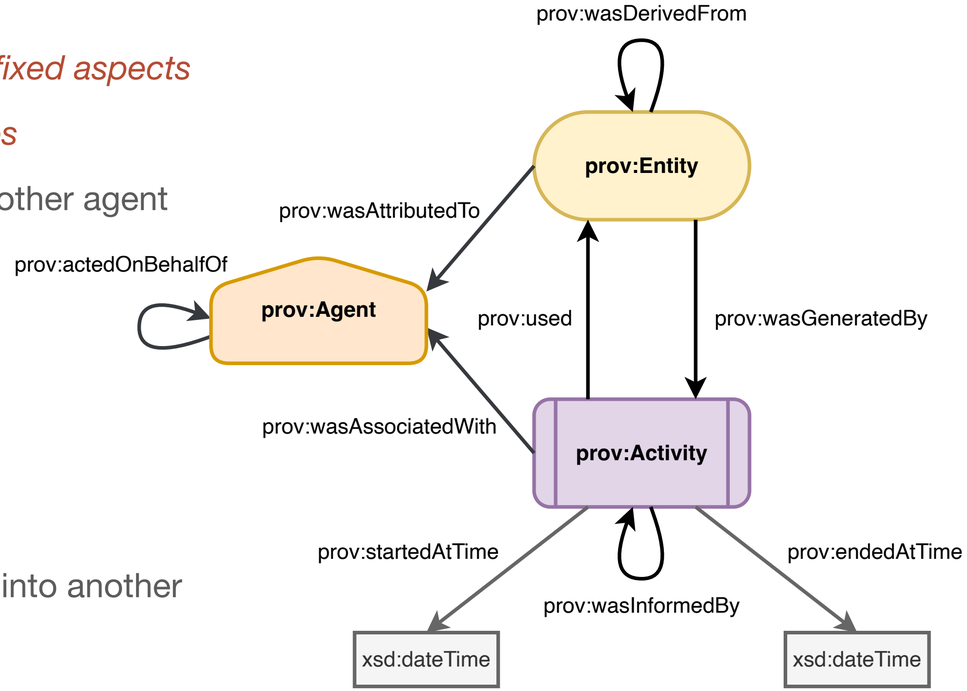


- *PROV-O*, a lightweight ontology, is suitable for a variety of applications
  - Represents provenance directly and *takes domain-specific details into account*, thus supporting *interoperable modeling*
  - Defines *classes*, *properties* and *rules* for representing and exchanging provenance between systems and contexts
    - Classifies terms into three categories for *different levels of detail* of provenance information:
      - *Starting point* terms, *Expanded* terms, and *Qualified terms*

Starting point terms

- The core of the PROV ontology comprises the *starting point classes and properties*
  - Used to create *simple provenance descriptions*

- **Starting point classes:**
  - prov:Entity - Physical, digital, conceptual or other real or imaginary *object with certain fixed aspects*
  - prov:Activity - Something that *happens over a period of time* and *acts on* or with *entities*
  - prov:Agent - *Responsible* for an *activity*, the existence of an *entity*, or the *actions* of another agent
- **Starting point properties:**
  - prov:startedAtTime - The *activity starts* at a certain point in time
  - prov:endedAtTime - The *activity ends* at a certain point in time
  - prov:used - *Use of an entity* by an activity
  - prov:wasGeneratedBy - The *entity is the result* of an activity
  - prov:wasDerivedFrom - Creating a *new*, *updating* an existing, *or transforming an entity* into another
  - prov:wasAttributedTo - Assigning an *entity to an agent*
  - prov:wasInformedBy - Provides a *dependency between* two *activities*
  - prov:wasAssociatedWith - *Delegation of responsibility* to the agent for the activity
  - prov:actedBehalfOf - *Chaining of responsibility* for activities and entities between agents



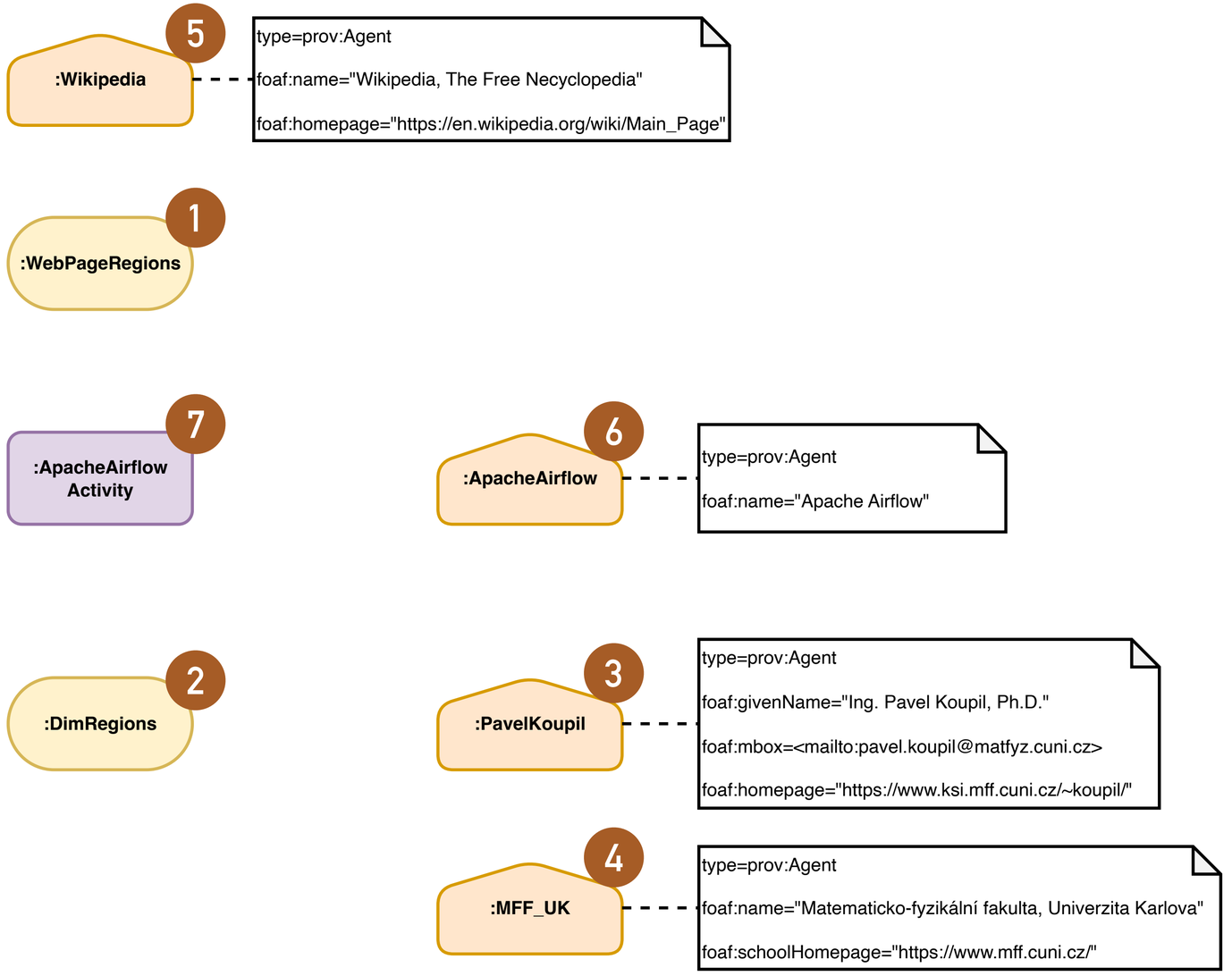
Example 9.1: Simple provenance description – ETL Regions

- Create an *illustration* of a provenance using the *starting point terms* from PROV-O
  - The illustration should describe the extraction, transformation and loading (ETL) of data from the Wikipedia page on Czech regions into the dim_regions table in the data warehouse (similarly to Example 6.4)

- Identify and *appropriately use* PROV-O *classes and properties*
  - Employ prov:Entity for each dataset / digital artefact
  - Apply prov:Agent for each person, organization, and script/software
  - Include prov:Activity for anything that occurs over a period of time and acts upon or with entities

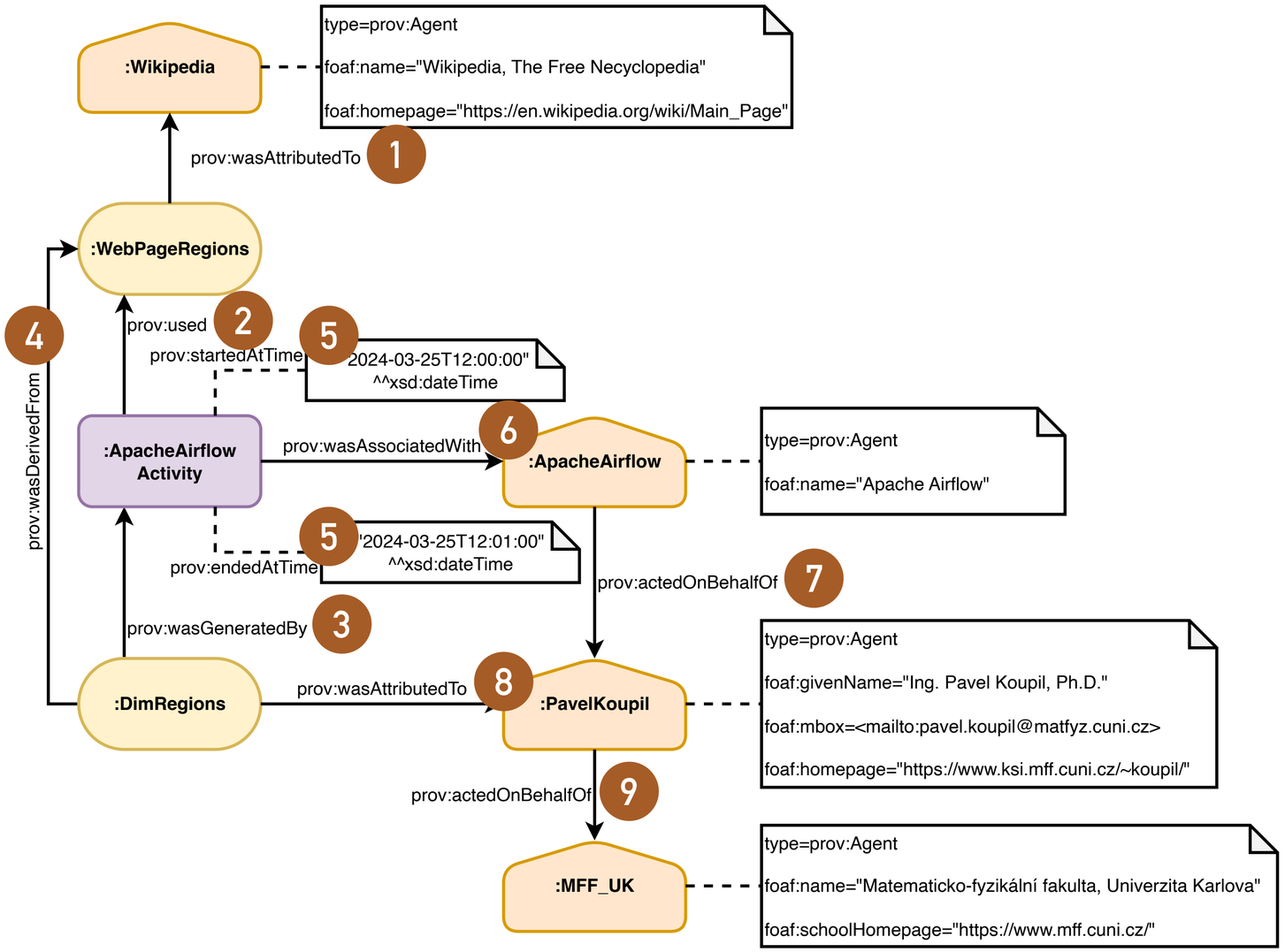
Example 9.1: Simple provenance description – ETL Regions (Solution)

- **Entities:**
  1. The *web content* of the Czech regions page is the *initial data artifact*
  2. The dim_regions table is the *target data artifact*
- **Agents:**
  3. *We are* one of the agents
  4. *MFF UK* is an *organization*
  5. *Wikipedia* is the organization that maintains the content of Wikipedia
  6. Apache Airflow is the *software used*
- **Activities:**
  7. *ETL workflow* implemented in Apache Airflow



Example 9.1: Simple provenance description – ETL Regions (Solution)

- **Properties:**
  1. Wikipedia is *responsible for web* content
  2. ETL workflow *works with* (i.e., uses) web resource
  3. The table *was generated by* ETL workflow
  4. The table *is derived from* data on the web
  5. ETL workflow *has a beginning and an end* (in time)
  6. Apache Airflow *performs* ETL workflow
  7. Apache Airflow performs activities *based on our requests*
  8. We are *responsible* for the outcome
  9. Our *superior organization* is the MFF UK



Example 9.2: Simple provenance description using Python

- Create a *Python script* that *produces a provenance document* using the *starting point terms* from PROV-O
  - The document should describe the extraction, transformation and loading (ETL) of data from the Wikipedia page on Czech regions into the dim_regions table in the data warehouse

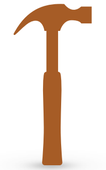
- Identify entities, agents, activities, and their properties
  - Employ prov:Entity for each dataset / digital artefact
  - Employ prov:Agent for each person, organization, and script/software
  - Include prov:Activity for anything that occurs over a period of time and acts upon or with entities

- *Save* the provenance document *as RDF TriG file*

Tip: To create an RDF representation, use the rdflib library

- (see https://rdflib.readthedocs.io/en/stable/)

Example 9.2: Simple provenance description using Python (Solution)



```python
from rdflib import Graph, Literal, Namespace, URIRef
from rdflib.namespace import RDF, FOAF, XSD, PROV
import logging, sys

NS = Namespace("https://ksi.mff.cuni.cz/~koupil/232-NDBI046#")

def create_prov_data() -> Graph:
  pass

def create_entities(collector: Graph) -> None:
  pass

def create_agents(collector: Graph) -> None:
  pass

def create_activities(collector: Graph) -> None:
  pass
```

import of necessary libraries

definition of namespace

decomposition of the assignment into individual tasks

Example 9.2: Simple provenance description using Python (Solution)

```python
def create_prov_data() -> Graph:
    result = Graph()
    result.bind("", NS)
    result.bind("rdf", RDF)
    result.bind("xsd", XSD)

    create_entities(result)
    create_agents(result)
    create_activities(result)

    return result
```

bind namespaces with prefixes

```python
def create_entities(collector: Graph) -> None:
    webpage_regions = NS.WebPageRegions
    collector.add((webpage_regions, RDF.type, PROV.Entity))
    collector.add((webpage_regions, PROV.wasAttributedTo, NS.Wikipedia))

    dim_regions = NS.DimRegions
    collector.add((dim_regions, RDF.type, PROV.Entity))
    collector.add((dim_regions, PROV.wasGeneratedBy, NS.ApacheAirflowActivity))
    collector.add((dim_regions, PROV.wasDerivedFrom, NS.WebPageRegions))
    collector.add((dim_regions, PROV.wasAttributedTo, NS.PavelKoupil))
```

specify the initial data artifact (i.e., entity)

the assigned agent will be created

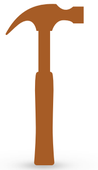
specify the target data artifact

as a result of the ETL workflow

derived from web source

responsibility for the outcome

Example 9.2: Simple provenance description using Python (Solution)



```
39 def create_agents(collector: Graph) -> None:
40   apache_airflow = NS.ApacheAirflow
41   collector.add((apache_airflow, RDF.type, PROV.Agent))
42   collector.add((apache_airflow, PROV.actedOnBehalfOf, NS.PavelKoupil))
43   collector.add((apache_airflow, FOAF.name, Literal("Apache Airflow", lang="en")))
44
45   author = NS.PavelKoupil
46   collector.add((author, RDF.type, PROV.Agent))
47   collector.add((author, PROV.actedOnBehalfOf, NS.MFF_UK))
48   collector.add((author, FOAF.givenName, Literal("Ing. Pavel Koupil, Ph.D.", lang="cs")))
49   collector.add((author, FOAF.mbox, URIRef("mailto:pavel.koupil@matfyz.cuni.cz")))
50   collector.add((author, FOAF.homepage, Literal("https://.../~koupil/", datatype=XSD.anyURI)))
51
52   organization = NS.MFF_UK
53   collector.add((organization, RDF.type, PROV.Agent))
54   collector.add((organization, FOAF.name, Literal("MFF, UK", lang="cs")))
55   collector.add((organization, FOAF.schoolHomepage, Literal("https...", datatype=XSD.anyURI)))
56
57   wikipedia = NS.Wikipedia
58   collector.add((wikipedia, RDF.type, PROV.Agent))
59   collector.add((wikipedia, FOAF.name, Literal("Wikipedia", lang="en")))
60   collector.add((wikipedia, FOAF.homepage, Literal("https://...", datatype=XSD.anyURI)))
```

definition of agents

specify a resource label

chaining of responsibility

specify additional attributes

Example 9.2: Simple provenance description using Python (Solution)

```
61 def create_activities(collector: Graph) -> None:
62   airflow_activity = NS.ApacheAirflowActivity
63   collector.add((airflow_activity, RDF.type, PROV.Activity))
64   collector.add((airflow_activity, PROV.startedAtTime,
       Literal("2024-03-25T12:00:00", datatype=XSD.dateTime)))
65   collector.add((airflow_activity, PROV.endedAtTime,
       Literal("2024-03-25T12:01:00", datatype=XSD.dateTime)))
66   collector.add((airflow_activity, PROV.used, NS.WebPageRegions))
67   collector.add((airflow_activity, PROV.wasAssociatedWith, NS.ApacheAirflow))
```

definition of activity

an activity has a beginning and an end

input data artifact

the agent responsible for performing the activity

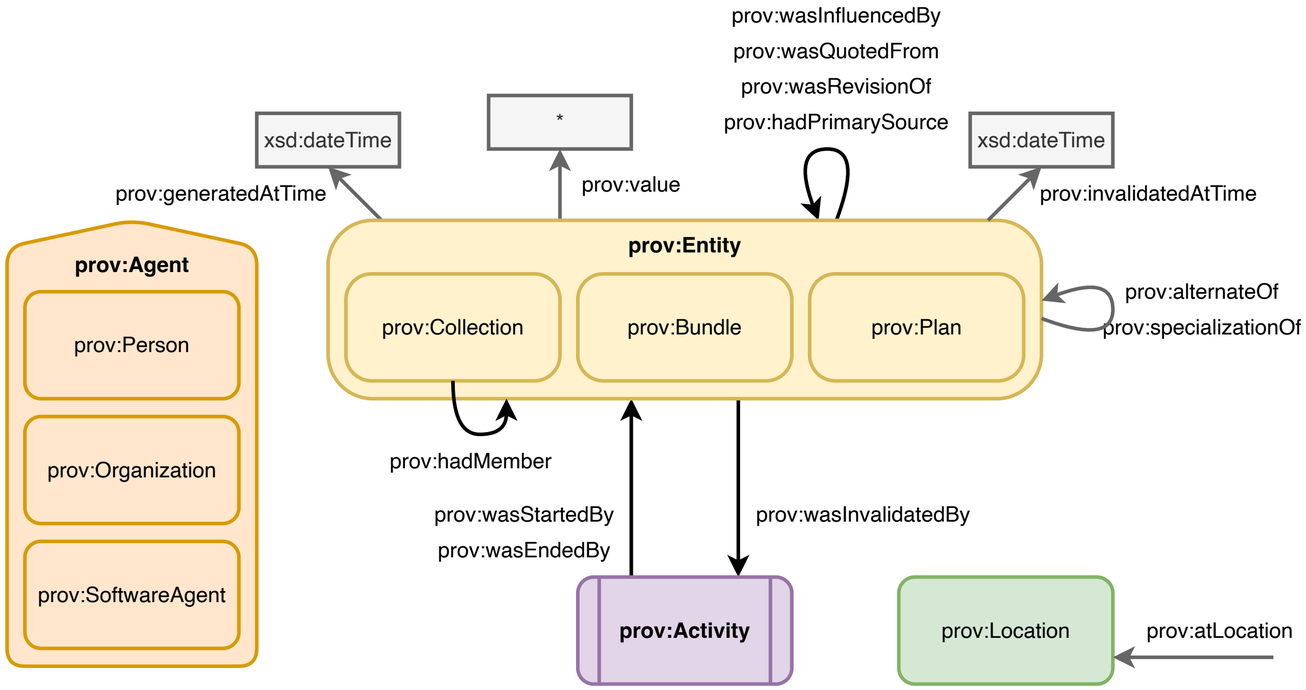
```
68 if __name__ == "__main__":
69   if len(sys.argv) != 2:
70     logging.error("Usage: python script.py <output_file_path>")
71     sys.exit(1)
72
73   create_prov_data().serialize(format="trig", destination=sys.argv[1]))
```

write the provenance document to the trig file

Expanded terms

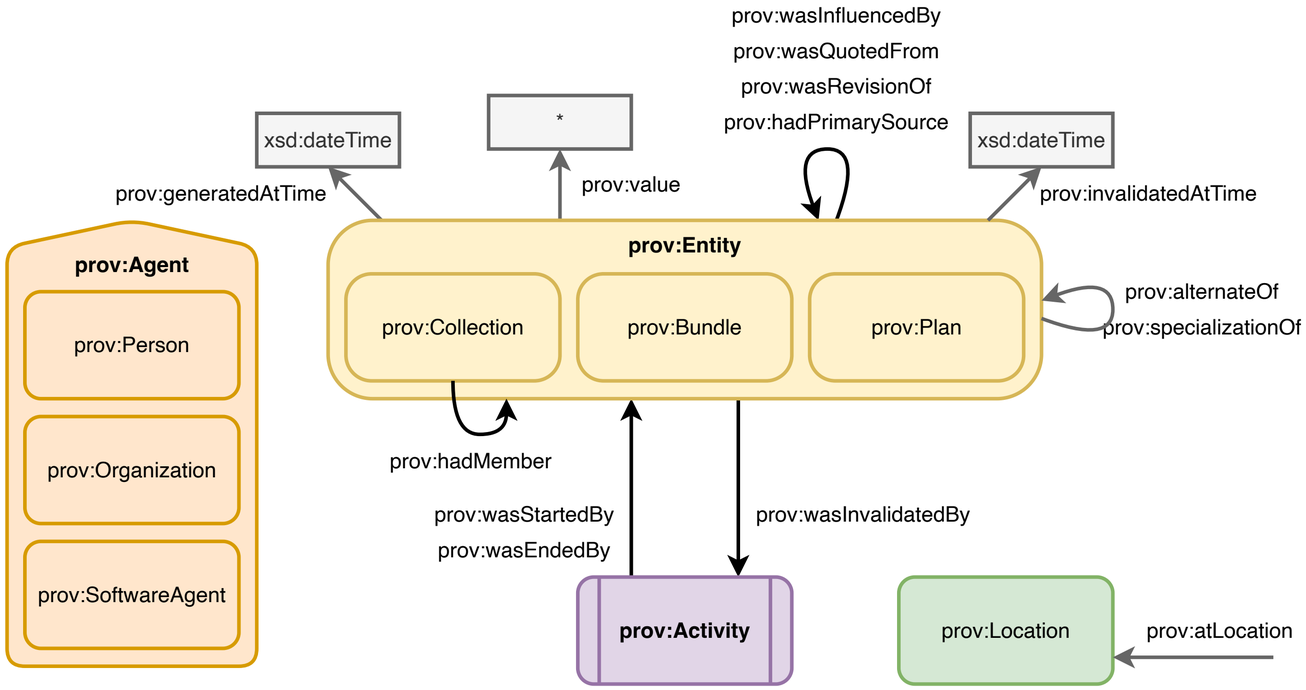
- Provides *additional terms* that can be used to link classes in the Starting Point category
  - For example, *subclasses* or *subproperties* of the Starting Point terms

- **Expanded classes:**
  - prov:Collection - An entity that *provides structure* to some components that are themselves entities
  - prov:EmptyCollection - Collection without members
  - prov:Bundle - Named set of provenance descriptions
  - prov:Person - Agent representing *people*
  - prov:SoftwareAgent - Agent representing *running software*
  - prov:Organization - Agent representing a *social or legal institution* such as a company, corporation, etc.
  - prov:Location - An identifiable *geographic location*, or also a *directory*, *row*, or *column*



Expanded terms

- **Expanded properties:**
  - prov:SpecializationOf - The *entity inherits* all aspects of its parent *and adds* its own *specificity*
  - prov:hadPrimarySource - A *primary source from an agent* with direct knowledge, created while studying a topic, without hindsight
  - prov:value - Provides a value that is a direct representation of the entity
  - prov:wasQuotedFrom - *Repetition of an entity* by someone who is probably not its original author
  - prov:wasStartedBy - The *activity is initiated* by the trigger that marks its start; it did not exist before this trigger
  - prov:wasEndedBy - A trigger indicating the *end of an activity* and terminating its existence
  - prov:atLocation - Location assignment
  - prov:generated - An *entity emerges from the activity*, the entity did not exist before and is now accessible



For a complete list of classes and properties, see the documentation: https://www.w3.org/TR/prov-o/#cross-reference-expanded-terms

Exercise 9.3: Utilizing expanded terms

- Create a *Python script* that *produces a provenance document* utilizing also the *expanded terms* from PROV-O
  - The document should describe a *complete ETL workflow implemented in Apache Airflow* (see Exercise 6.6)

- Identify and distinguish entities, agents, activities, and their properties using the expanded terms
  - Employ prov:Entity for each dataset / digital artefact
  - Employ prov:Person, prov:Organization, prov:SoftwareAgent for each person, organization, and script/software respectively
  - Include prov:Activity for anything that occurs over a period of time and acts upon or with entities
  - Distinguish additional properties, e.g., prov:hadPrimarySource, prov:wasInfluencedBy, prov:atLocation

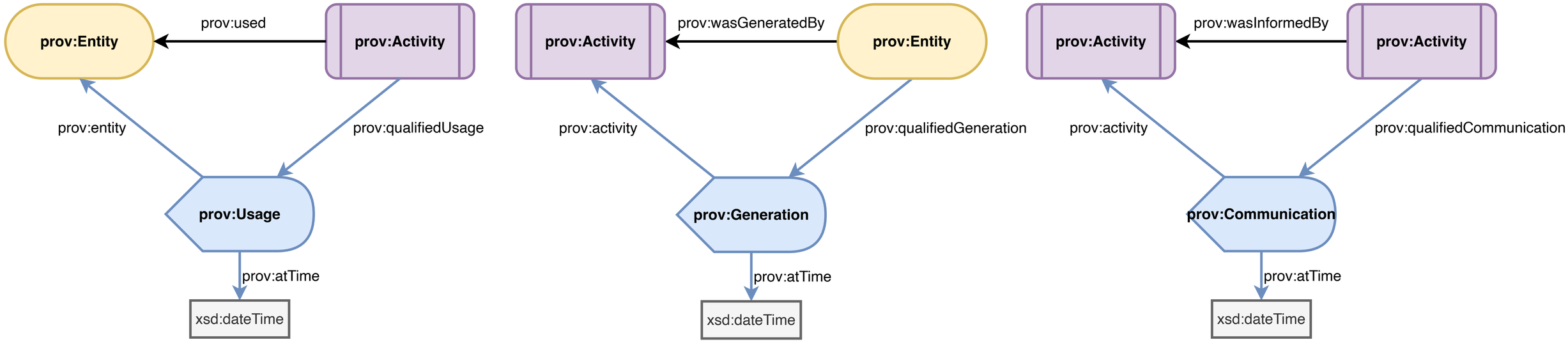
- *Save* the provenance document *as RDF TriG file*

Tip: To create an RDF representation, use the rdflib library

- (see https://rdflib.readthedocs.io/en/stable/)

Qualified terms

- Provide *additional classes and properties* for binary relations that are otherwise composed of Starting point and Extended terms
  - The terms in this category follow a pattern that differs from the patterns in the Starting Point and Extended categories



For a complete list of classes and properties, see the documentation: https://www.w3.org/TR/prov-o/#cross-reference-qualified-terms

Exercise 9.4: Extending provenance document utilizing qualified terms

- Refer to the *PROV-O documentation* and create a *Python script* that produces a provenance document *using also qualified terms*
  - Extend on the solution to Exercise 9.3, describing in particular the *creation of Tableau visualizations*:
    - Bar chart (see Examples 4.6 and 4.7)
    - Dual axis chart (see Example 4.9)
    - Scatter plot (see Example 4.10)
    - Pareto chart (see Example 4.11)

- Identify and distinguish additional entities, agents, activities, and their properties
  - *Use* the starting point, expanded, and qualified *terms appropriately*

- *Save* the provenance document *as RDF TriG file*

Tip: To create an RDF representation, use the rdflib library

- (see https://rdflib.readthedocs.io/en/stable/)

Exercise 9.5: Extending provenance document utilizing qualified terms II

- Refer to the *PROV-O documentation* and create a *Python script* that produces a provenance document *utilizing also qualified terms*
  - Extend on the solution to Exercise 9.4, describing in particular the *creation of a data cube* (see Examples/Exercises 7.1 to 7.6) and its *validation using a Python script* (see Exercise 7.7)

- Identify and distinguish additional entities, agents, activities, and their properties
  - Use the starting point, expanded, and qualified terms appropriately

- *Save* the provenance document *as RDF TriG file*

Tip: To create an RDF representation, use the rdflib library

- (see https://rdflib.readthedocs.io/en/stable/)

References

Data provenance

- PROV-Overview: https://www.w3.org/TR/prov-overview/
- PROV-PRIMER: https://www.w3.org/TR/2013/NOTE-prov-primer-20130430/
- PROV-O: https://www.w3.org/TR/2013/REC-prov-o-20130430/
- PROV-DM: https://www.w3.org/TR/2013/REC-prov-dm-20130430/
- PROV-CONSTRAINTS: https://www.w3.org/TR/prov-constraints/

Python

- rdflib: https://rdflib.readthedocs.io/en/stable/