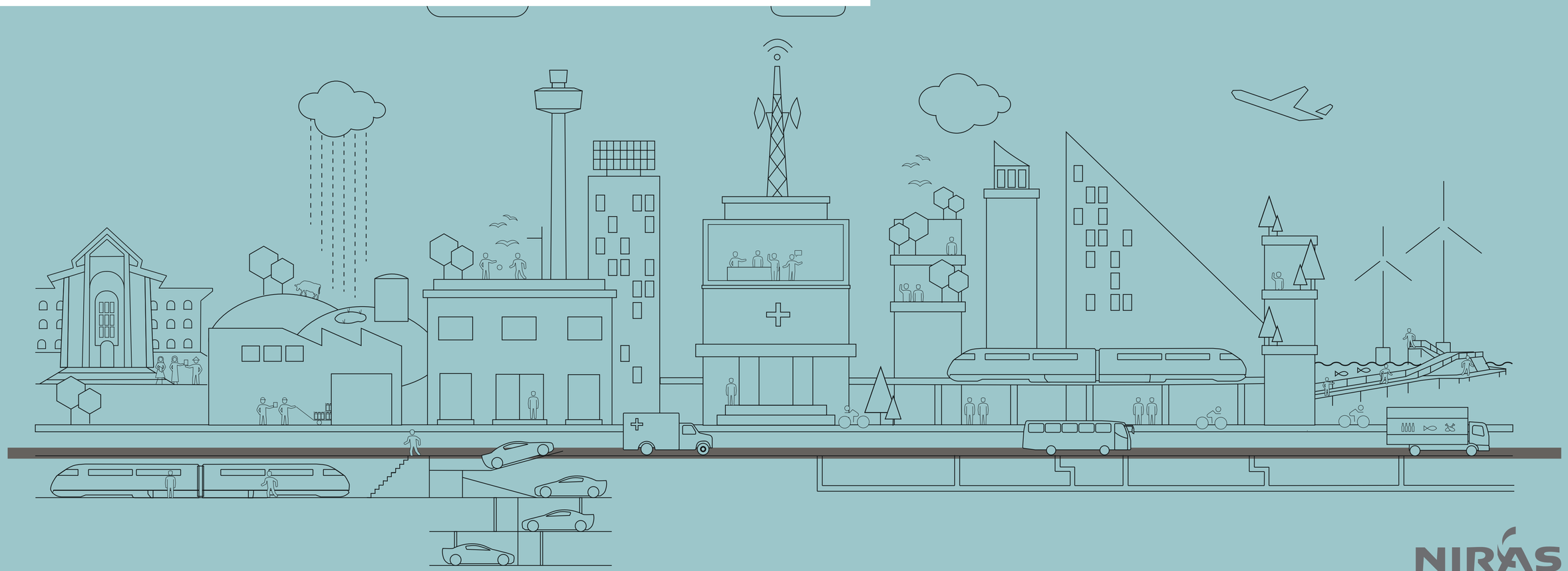
OPM

Ontology for Property Management

Mads Holten Rasmussen[1,2] | M Bonduel[3] | M Lefrançois[4] | C A Hviid[1] | J Karlshøj[1]

NIRAS[1] | DTU[2] | KU Leuven[3] | MINES Saint-Étienne[4]

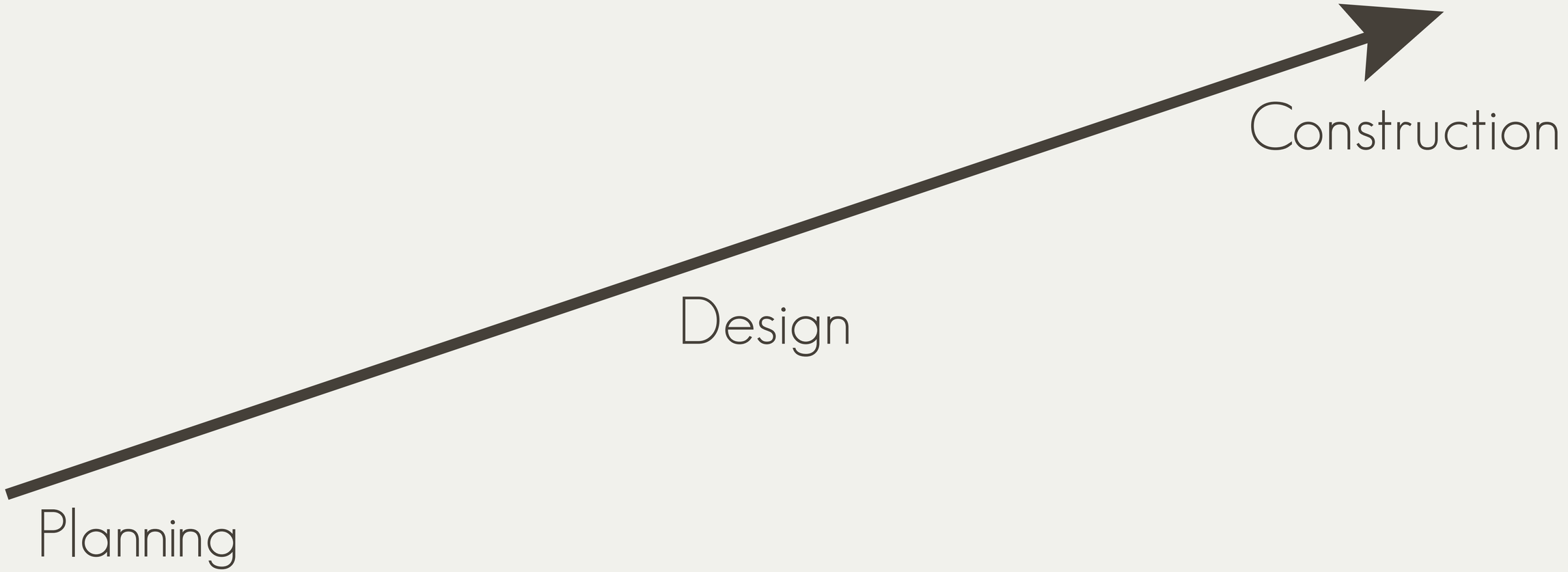
June 19th 2018



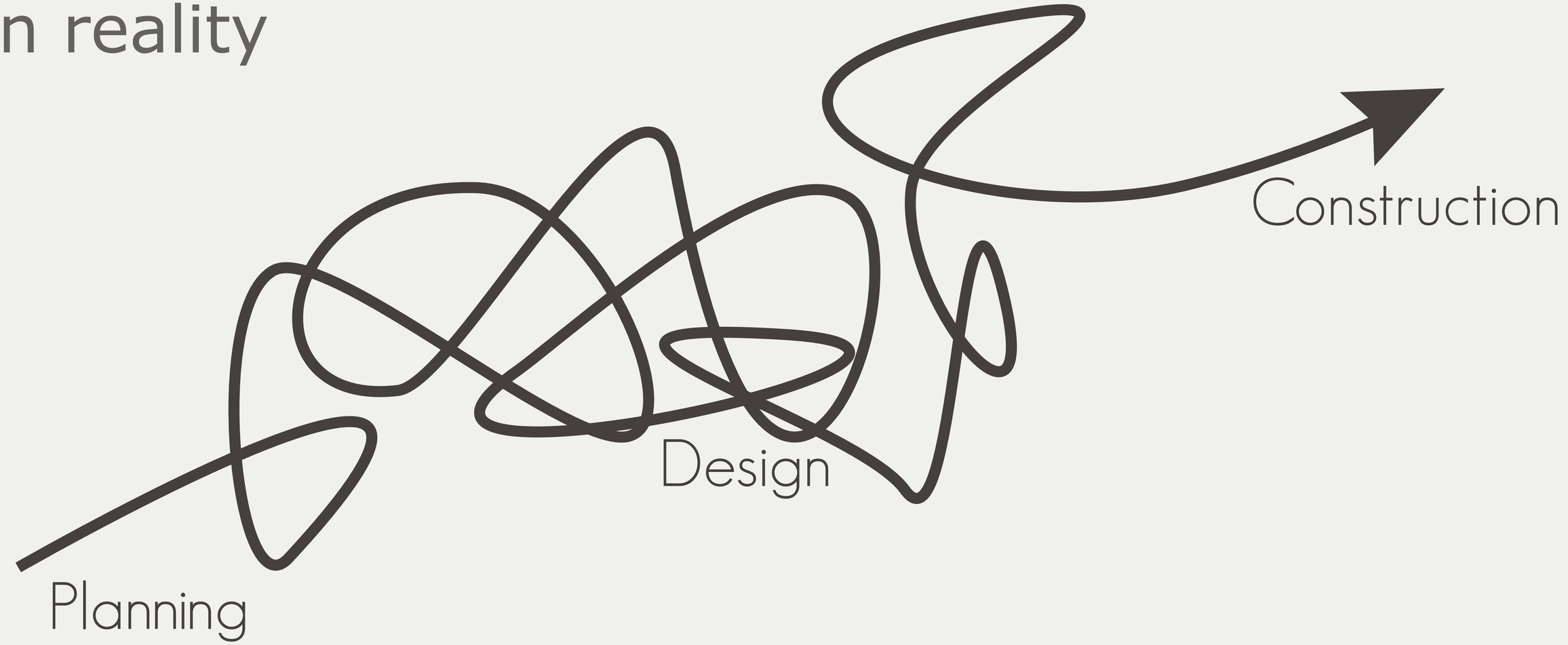
01

Background

Building design

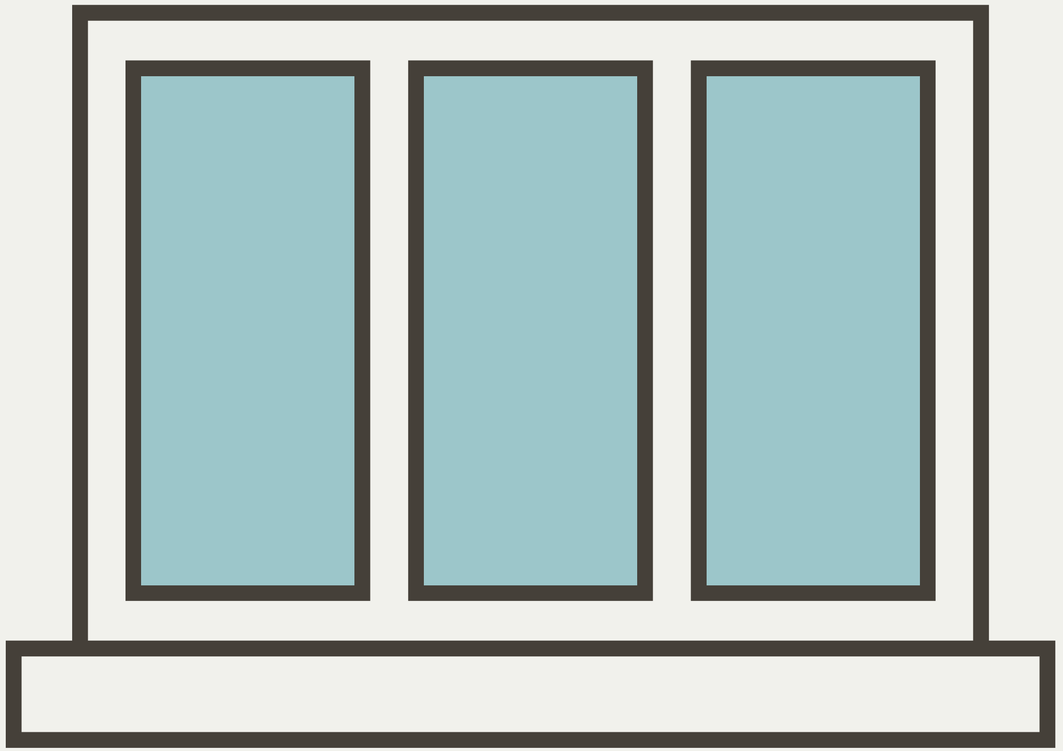


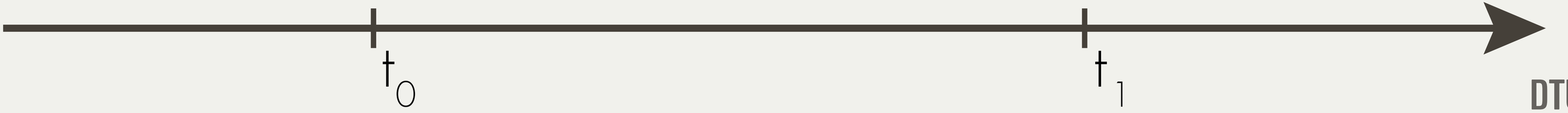
Building design in reality

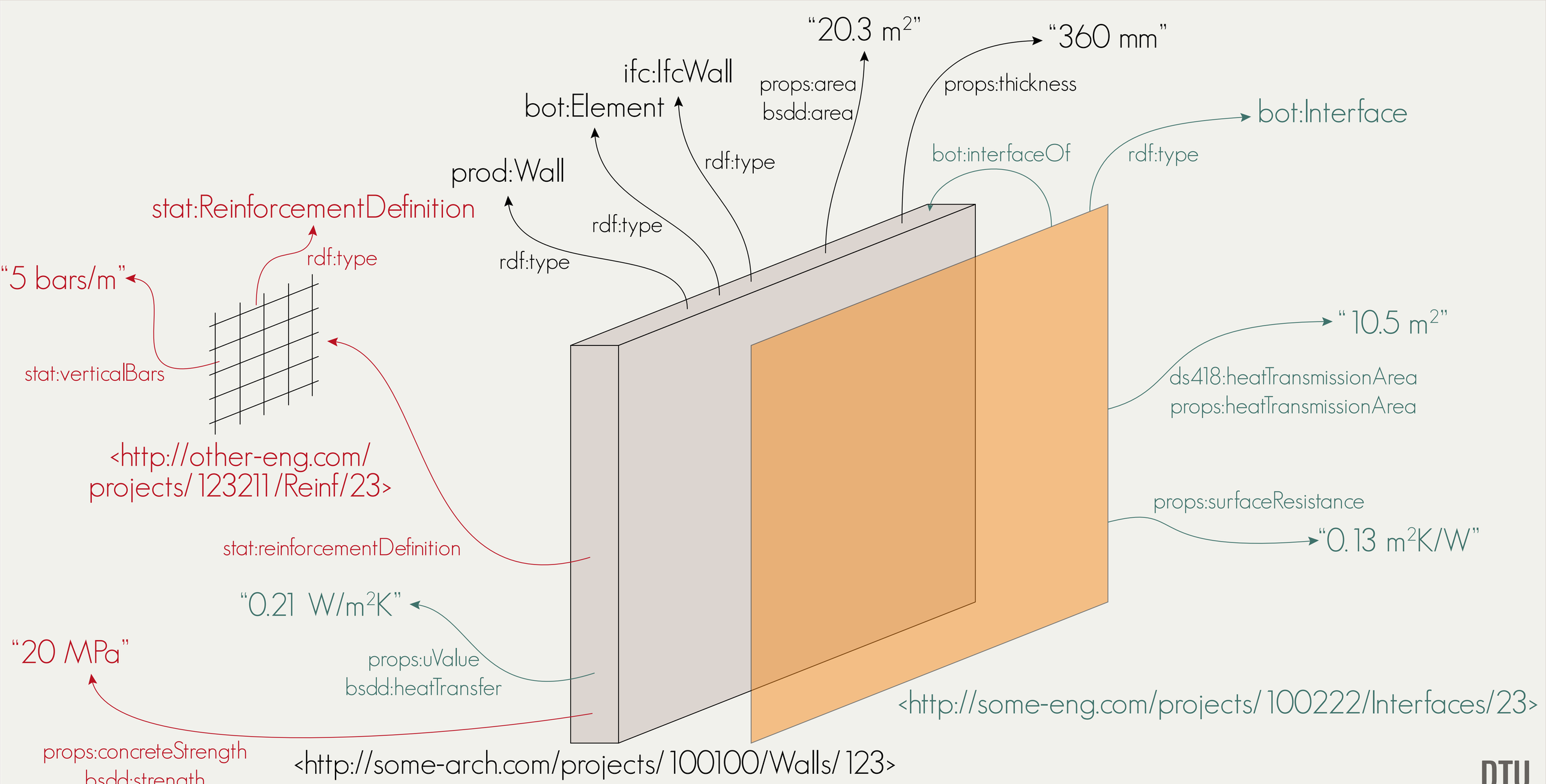


<Window-325>

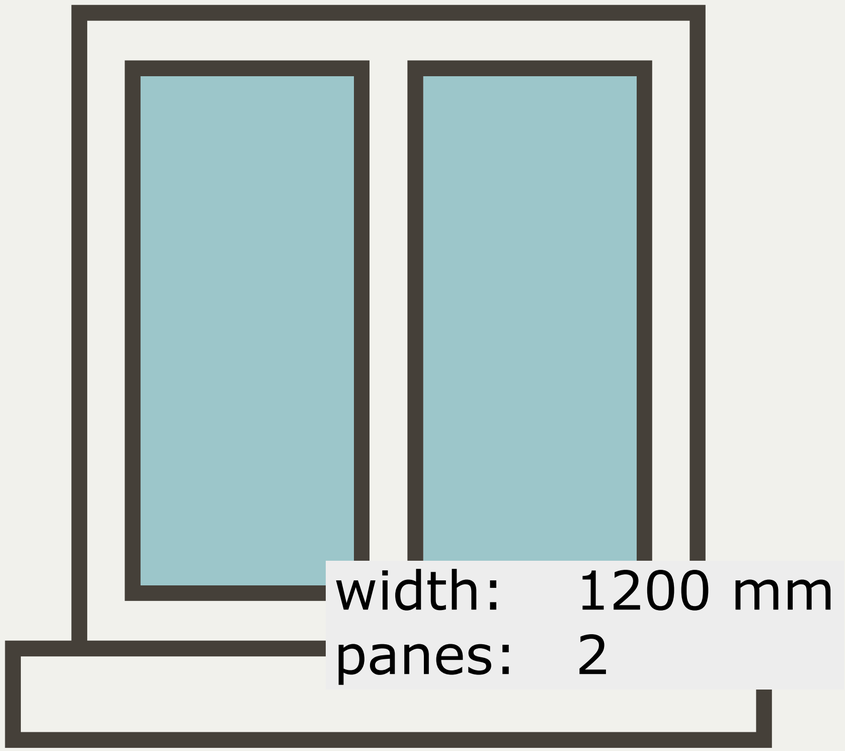
Still <Window-325> ???



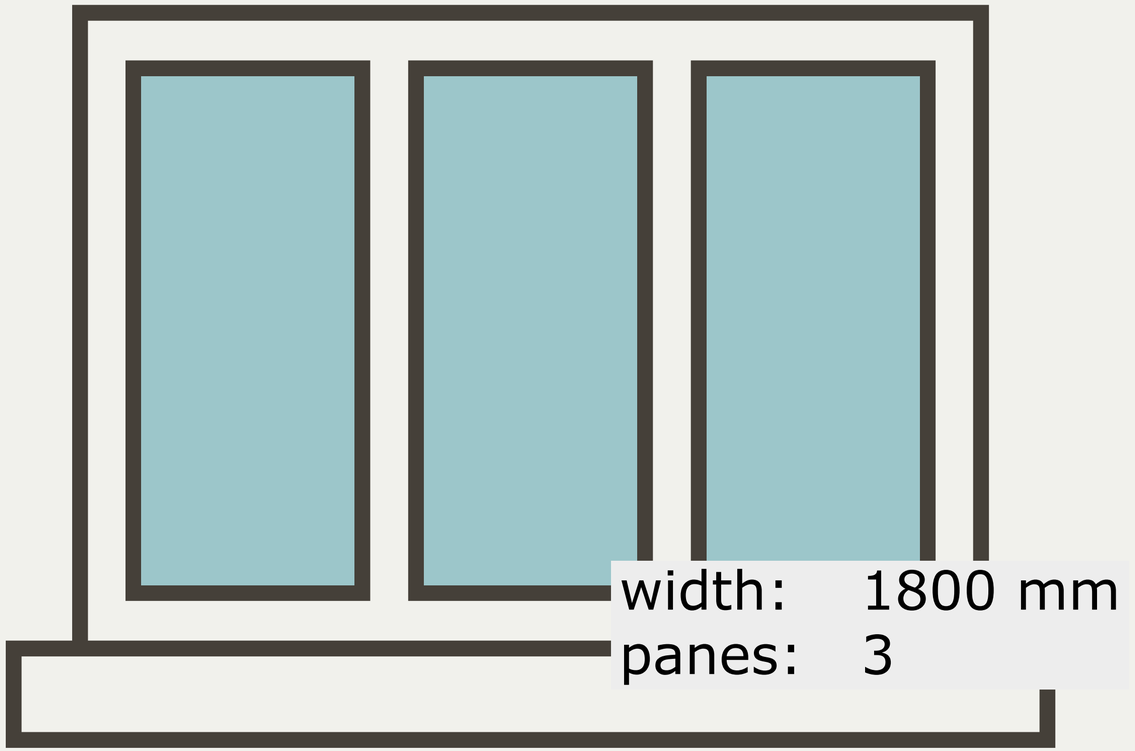



"20.3 m²"
"360 mm"
ifc:IfcWall
bot:Element
prod:Wall
rdf:type
rdf:type
rdf:type
props:area
bsdd:area
props:thickness
bot:Interface
bot:interfaceOf
rdf:type
stat:ReinforcementDefinition
rdf:type
"5 bars/m"
stat:verticalBars
<http://other-eng.com/projects/123211/Reinf/23>
"10.5 m²"
ds418:heatTransmissionArea
props:heatTransmissionArea
props:surfaceResistance
"0.13 m²K/W"
stat:reinforcementDefinition
"0.21 W/m²K"
props:uValue
bsdd:heatTransfer
"20 MPa"
<http://some-eng.com/projects/100222/Interfaces/23>
props:concreteStrength
bsdd:strength
<http://some-arch.com/projects/100100/Walls/123>

<Window-325>



<Window-325>

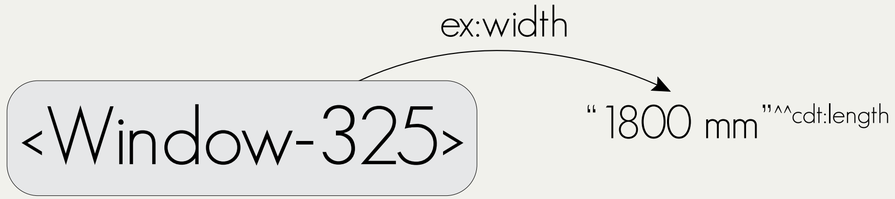


How to allow changes while still being able to maintain history?

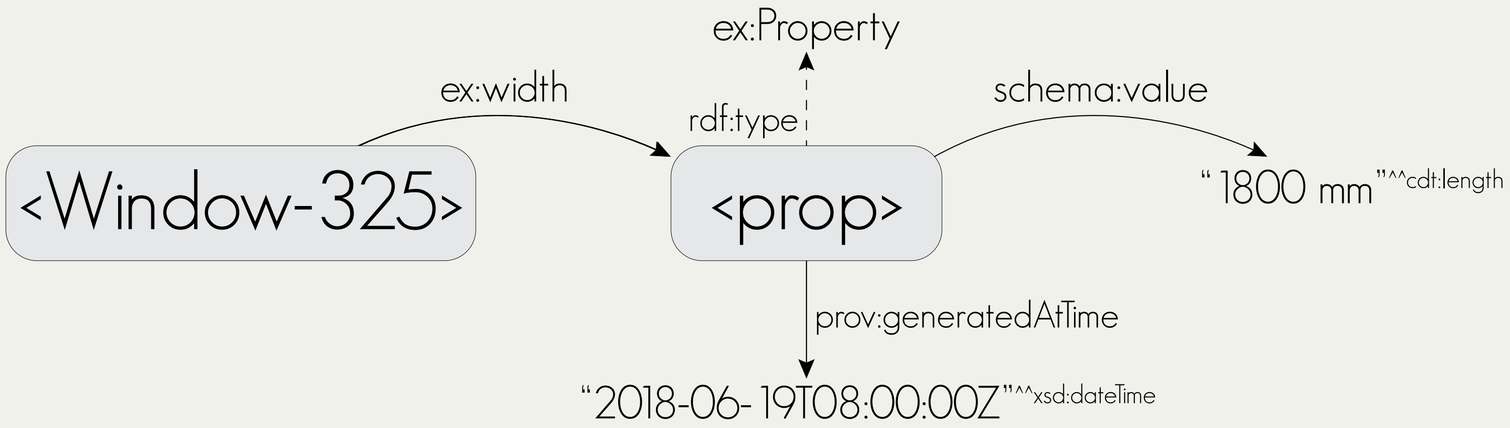
02

L1, L2, L3

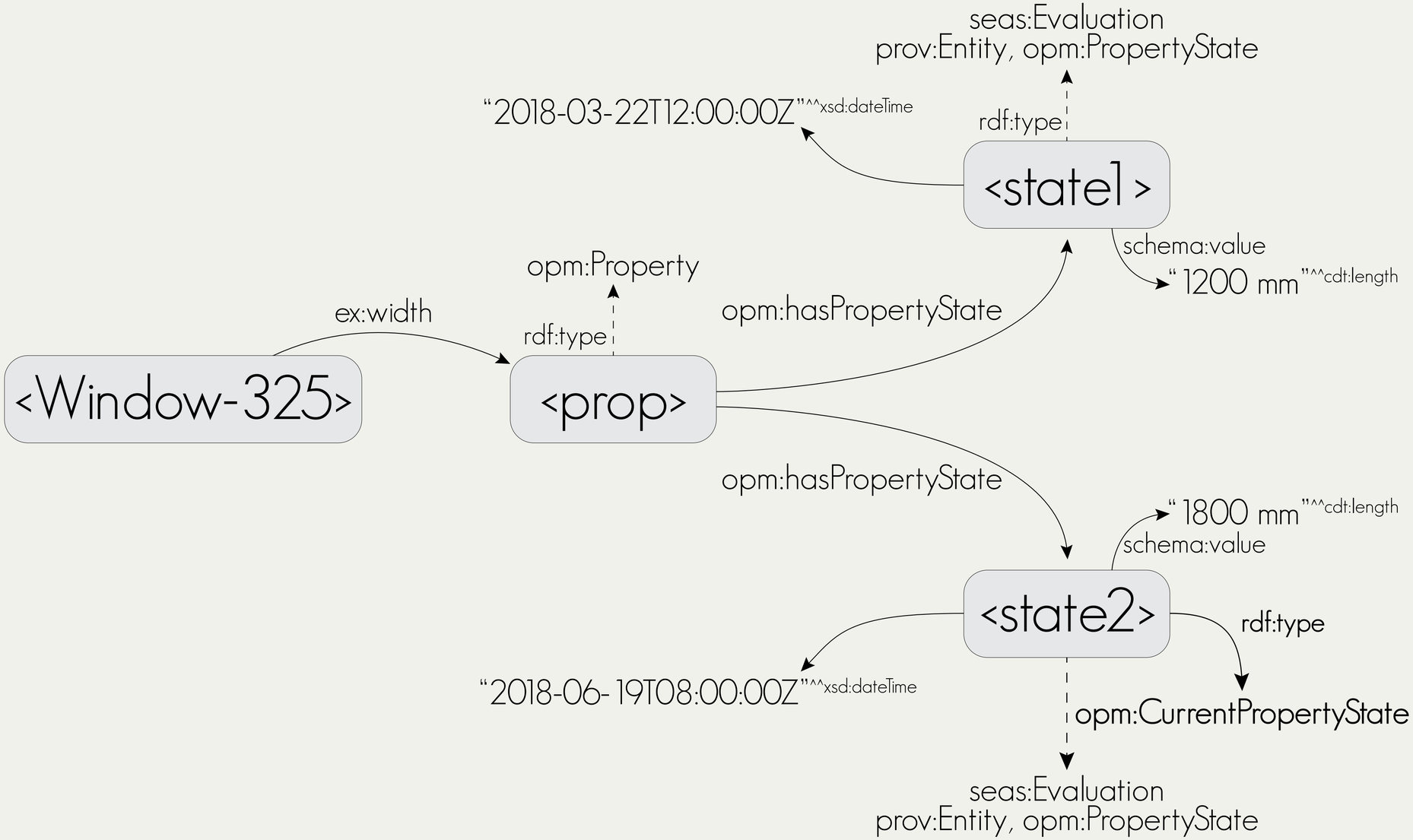
L1



L2


ex:Property
ex:width
rdf:type
schema:value
<Window-325>
<prop>
“1800 mm”^^cdt:length
prov:generatedAtTime
“2018-06-19T08:00:00Z”^^xsd:dateTime

L3


seas:Evaluation
prov:Entity, opm:PropertyState
"2018-03-22T12:00:00Z"^^xsd:dateTime
rdf:type
<state1>
schema:value
"1200 mm"^^cdt:length
opm:Property
ex:width
rdf:type
opm:hasPropertyState
<Window-325>
<prop>
opm:hasPropertyState
"1800 mm"^^cdt:length
schema:value
<state2>
rdf:type
"2018-06-19T08:00:00Z"^^xsd:dateTime
opm:CurrentPropertyState
seas:Evaluation
prov:Entity, opm:PropertyState

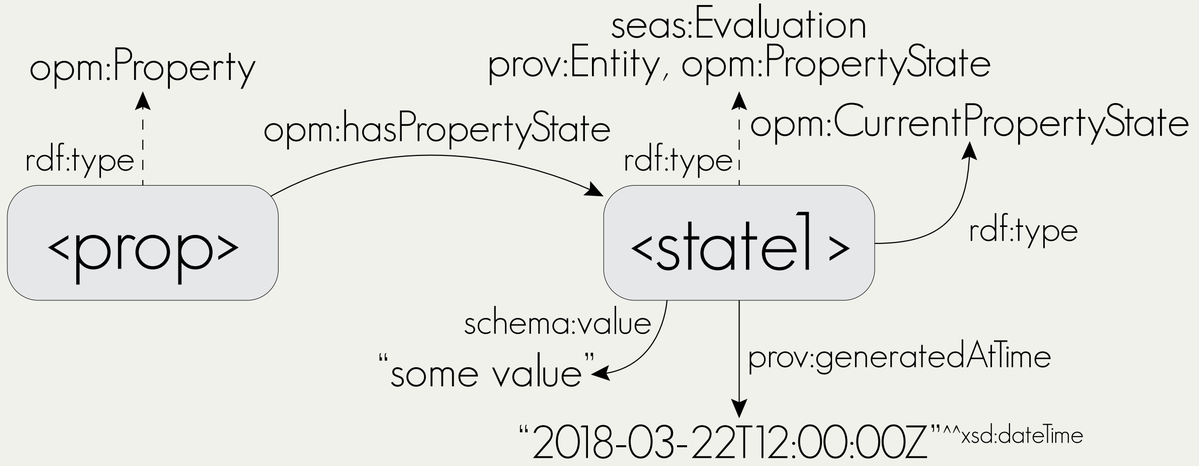
L1 vs. L2. vs. L3

	Units	Metadata	History
L1	✓ datatype	✗	✗
L2	✓	✓	✗
L3	✓	✓	✓

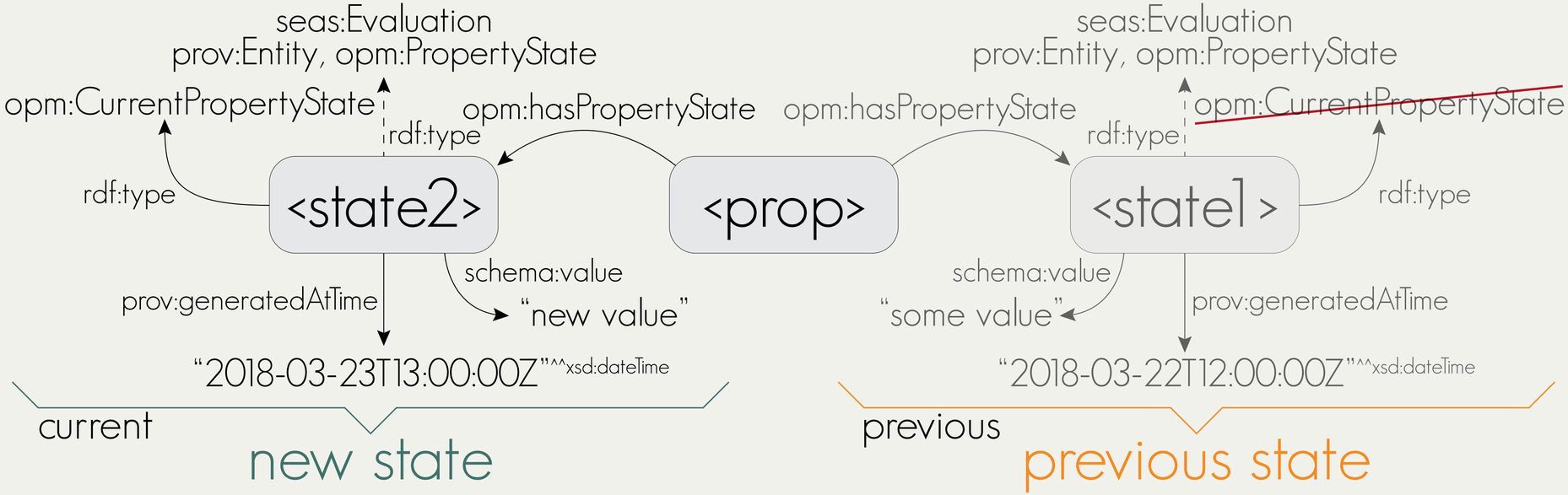
03

Using OPM

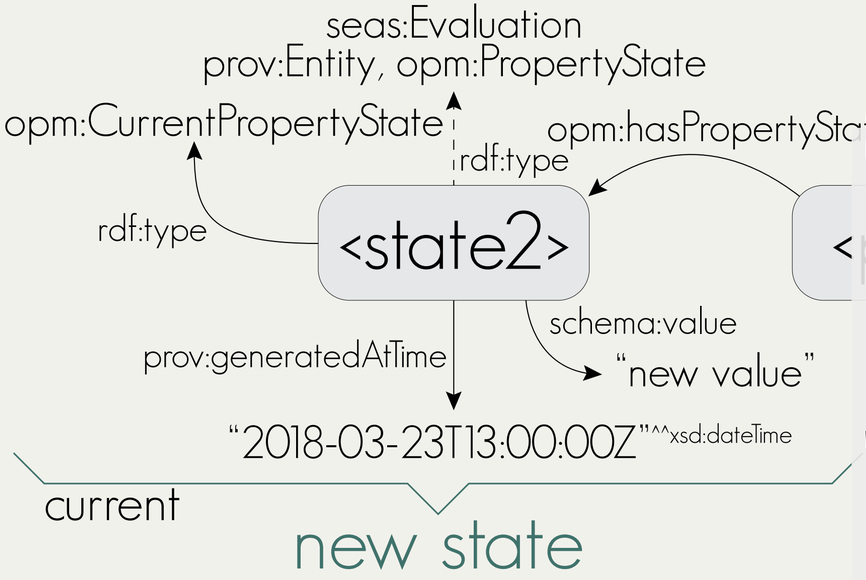
First state


seas:Evaluation
prov:Entity, opm:PropertyState
opm:Property
opm:hasPropertyState
opm:CurrentPropertyState
rdf:type
rdf:type
rdf:type
<prop>
<state1>
schema:value
"some value"
prov:generatedAtTime
"2018-03-22T12:00:00Z"^^xsd:dateTime

Second state

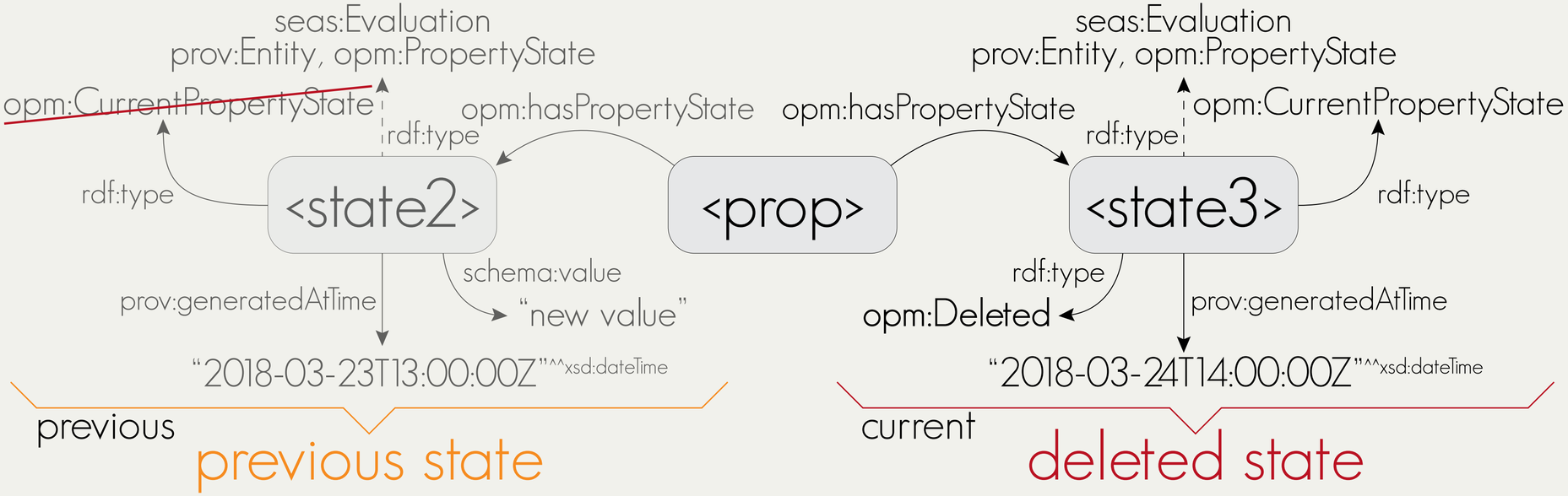


Second state

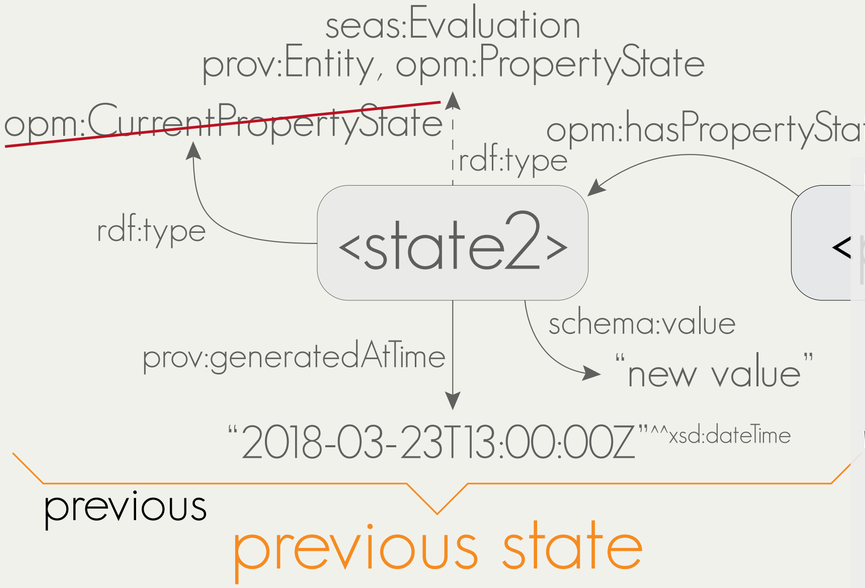


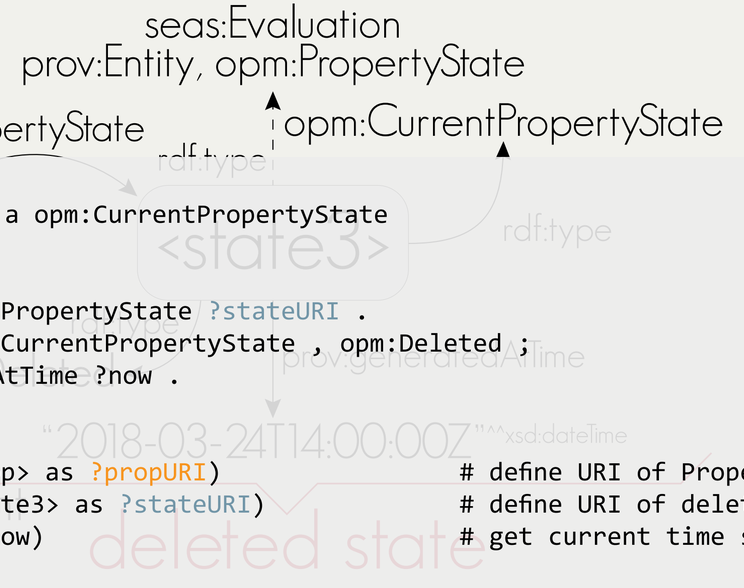

```
DELETE {
  ?previousState a opm:CurrentPropertyState
}
INSERT {
  ?propURI opm:hasPropertyState ?stateURI .
  ?stateURI a opm:CurrentPropertyState ;
    prov:generatedAtTime ?now ;
    schema:value ?val .
}
WHERE {
  BIND(<prop> as ?propURI)                  # define URI of Property
  BIND("new value" as ?val)                 # define new value
  BIND(<wall_A#state2> as ?stateURI)        # define URI for State
  BIND(NOW() as ?now)                       # get current time stamp

  ?propURI opm:hasPropertyState ?previousState .
  ?previousState a opm:CurrentPropertyState ;
    schema:value ?currentVal .              # get value of current state
  FILTER(?val != ?currentVal)               # don't update if equal to latest state
}
```

Third state (delete)



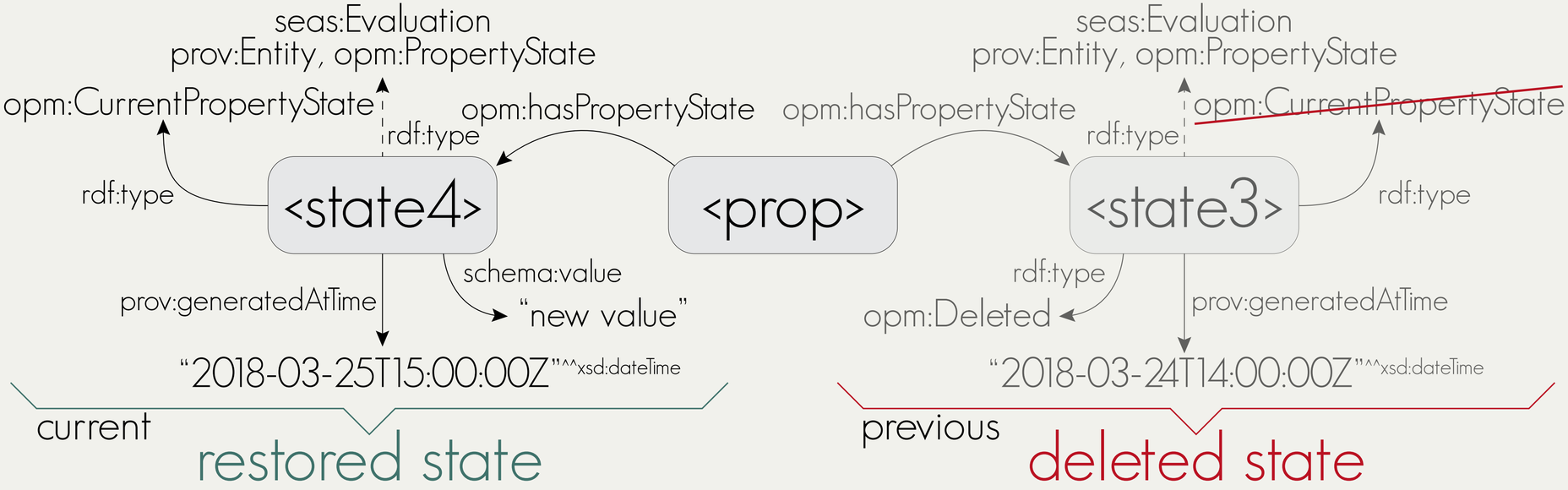
Third state (delete)



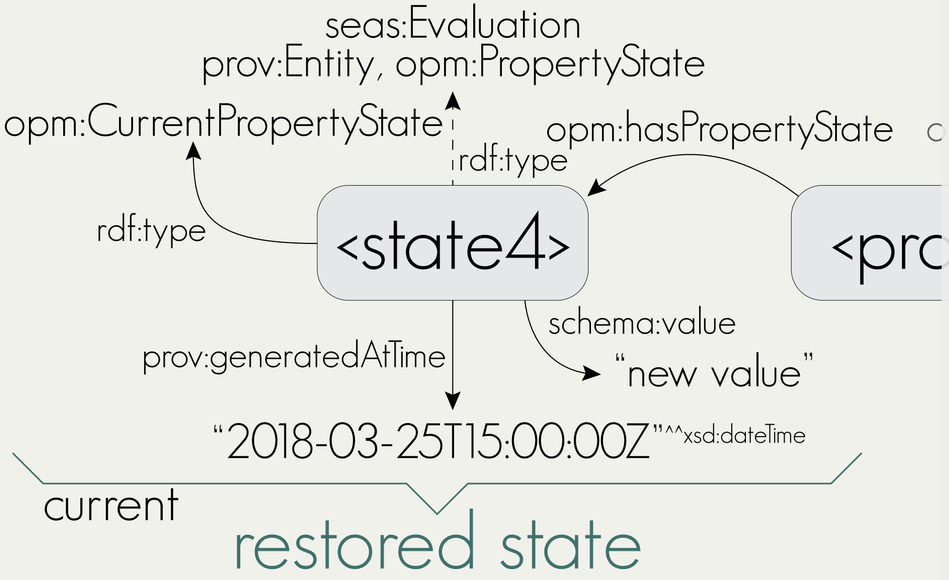



```
DELETE {
    ?previousState a opm:CurrentPropertyState
}
INSERT {
   ?propURI opm:hasPropertyState ?stateURI .
   ?stateURI a opm:CurrentPropertyState , opm:Deleted ;
    prov:generatedAtTime ?now .
}
WHERE {
  BIND(<wall_A#prop> as ?propURI)                  # define URI of Property
  BIND(<wall_A#state3> as ?stateURI)               # define URI of deleted State
  BIND(NOW() as ?now)                              # get current time stamp

  ?propURI opm:hasPropertyState ?previousState .
  ?previousState a opm:CurrentPropertyState .      # get current state

  # do not delete if the current state is already a opm:Deleted
  MINUS { ?previousState a opm:Deleted }
}
```

Fourth state (restore)

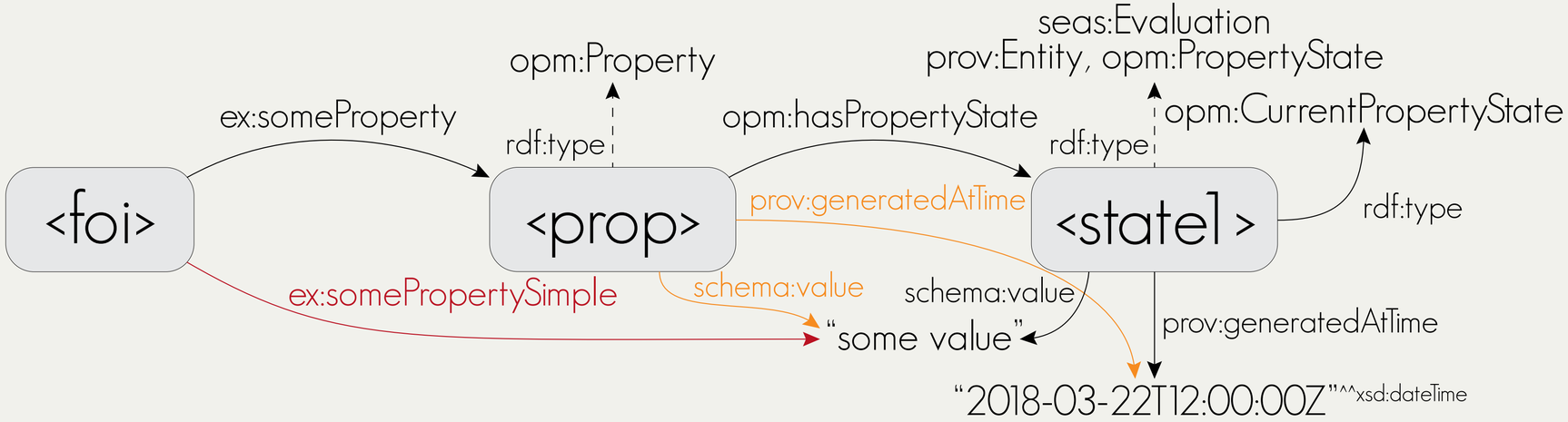


Fourth state (restore)




```
DELETE {
  ?previousState a opm:CurrentPropertyState
}
INSERT {
  ?propURI opm:hasPropertyState ?stateURI .
  ?stateURI a opm:CurrentPropertyState ;
    prov:generatedAtTime ?now ;
    ?key ?val .
}
WHERE {
  BIND(<wall_A#prop> as ?propURI)    # define URI of Property
  BIND(<wall_A#state4> as ?stateURI) # define URI of new State
  BIND(NOW() as ?now)                # get current time stamp

  # get time stamp of most recent property state that was not deleted
  { SELECT ?propURI (MAX(?time) AS ?t)
    WHERE {
      ?propURI opm:hasPropertyState ?s .
      ?s schema:value ?lastVal ;
         prov:generatedAtTime ?time .
      MINUS { ?s a opm:Deleted }
    } GROUP BY ?propURI }

  # get key-value pairs of latest state that is not deleted
  ?propURI opm:hasPropertyState [
    prov:generatedAtTime ?t ;
    ?key ?val ]
  FILTER(?key != prov:generatedAtTime) # filter out time stamps

  # get previous state
  ?propURI opm:hasPropertyState ?previousState .
  ?previousState a opm:CurrentPropertyState .
}
```

L1, L2, L3
in same graph

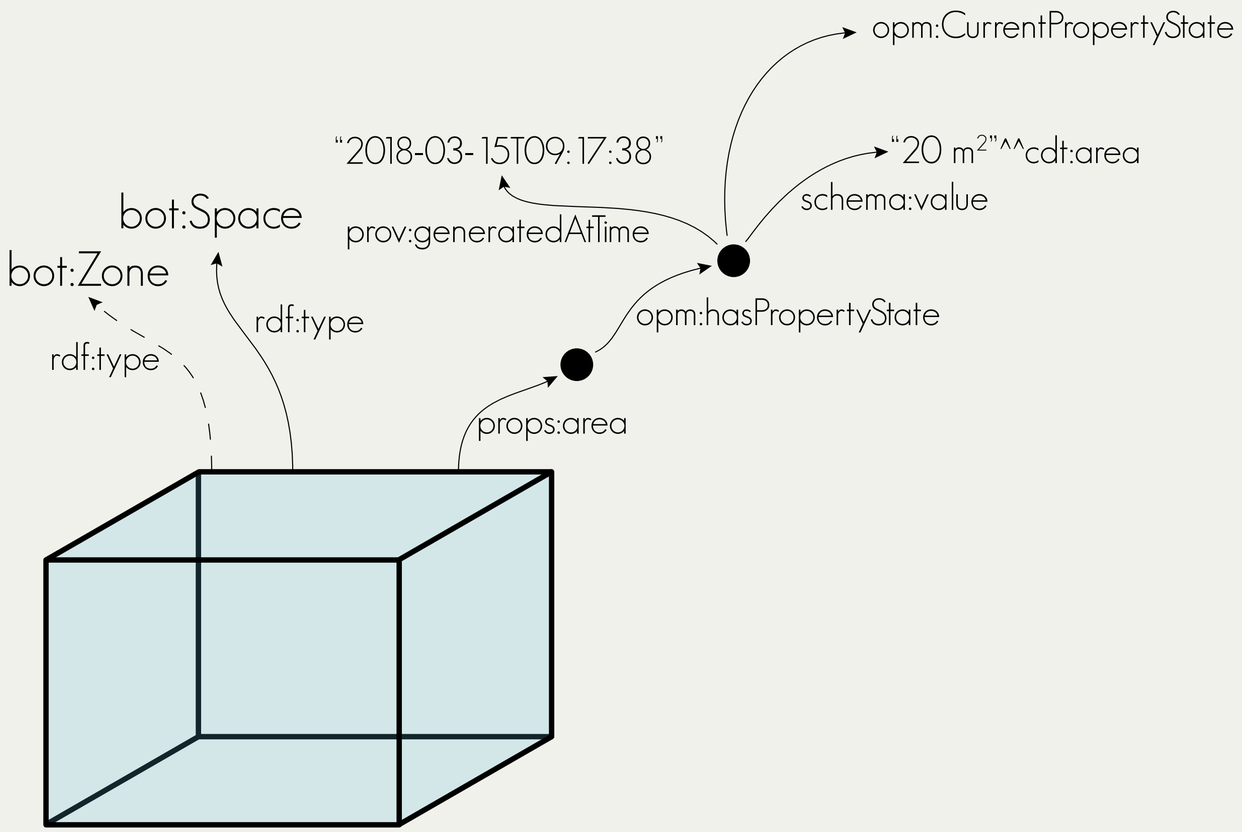


DEMO

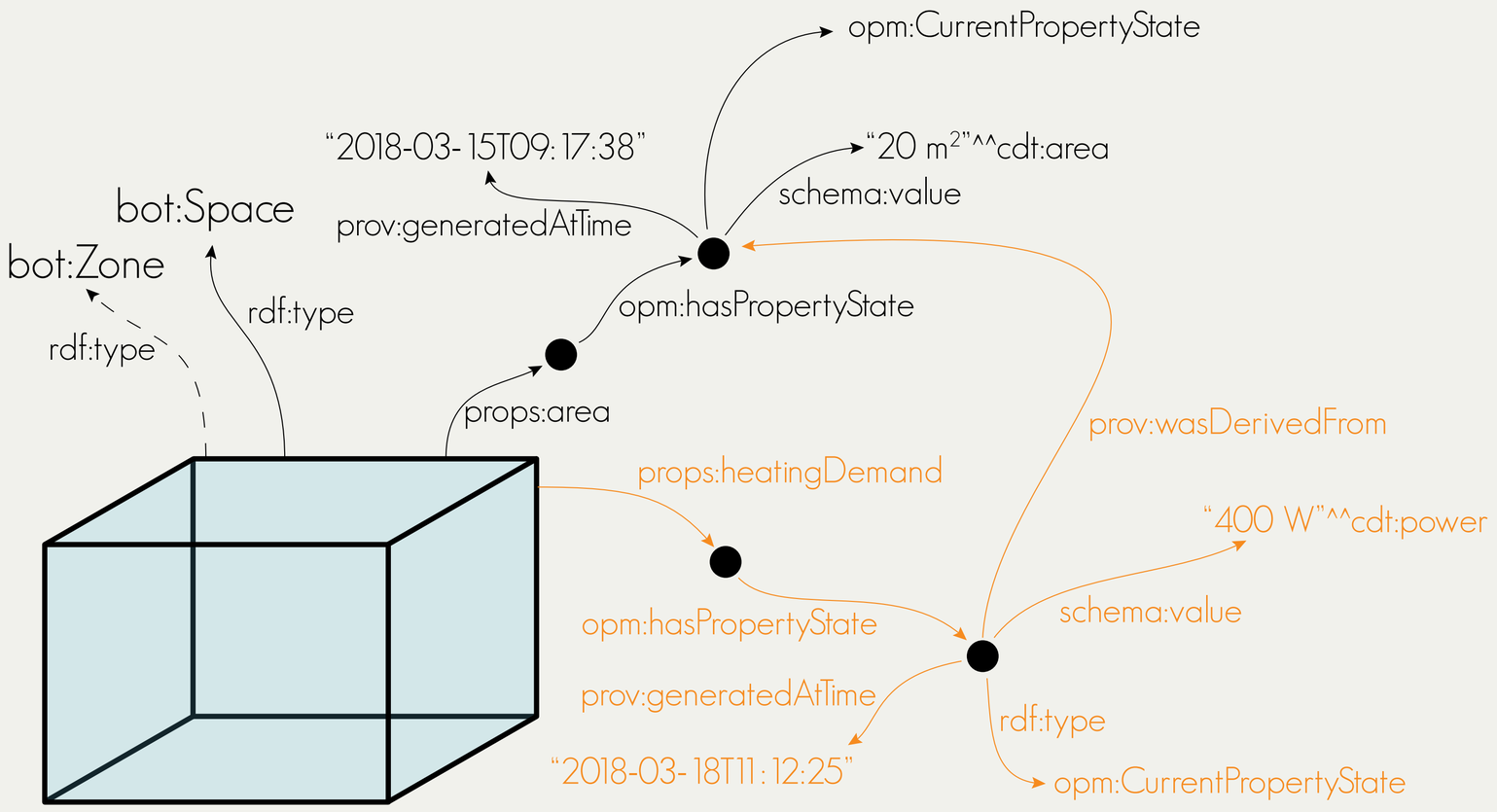
04

Proof of Concept

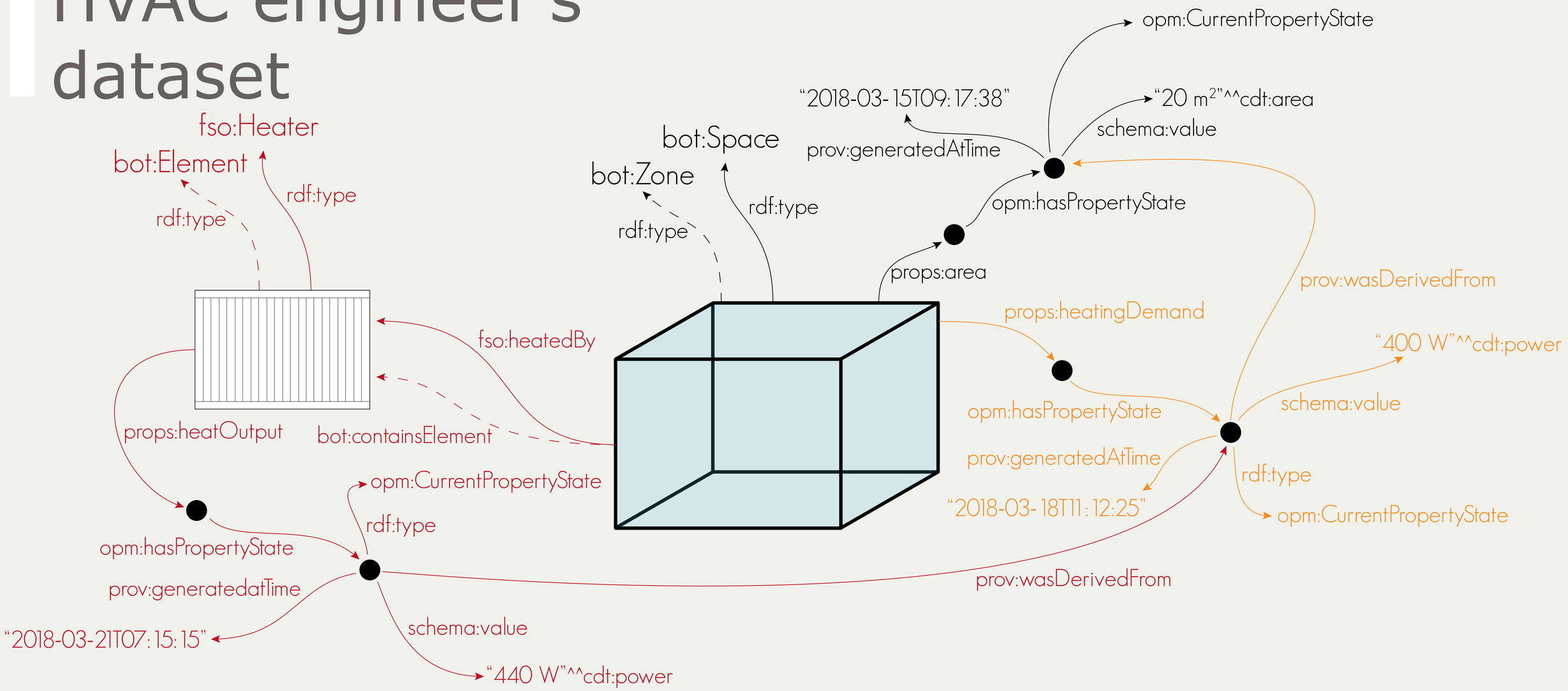
Architect's dataset



ICE engineer's dataset



HVAC engineer's dataset



DEMO

Implementations

Revit BOT Exporter	Mads Holten Rasmussen	https://github.com/MadsHolten/revit-bot-exporter
IFC-LBD	Jyrki Oraskari, Mathias Bonduel	https://github.com/jyrkioraskari/IFCtoLBD
Query Generator	Mads Holten Rasmussen	https://github.com/MadsHolten/opm-qg

The numbers

	This work	OPM Full
Classes	4	8
Object properties	1	5
Datatype properties	0	2
Logical axioms	7	19
DL Expressivity	$\mathcal{ALCH}$	$\mathcal{ALCH(D)}$

Conclusion

- there is a demand for tracking history during design of buildings

- possible to store every state of a property

- possible to relate a derived property to the very state of the argument it is based on

- future work on derived properties
  - consequence analysis
  - who is influenced?