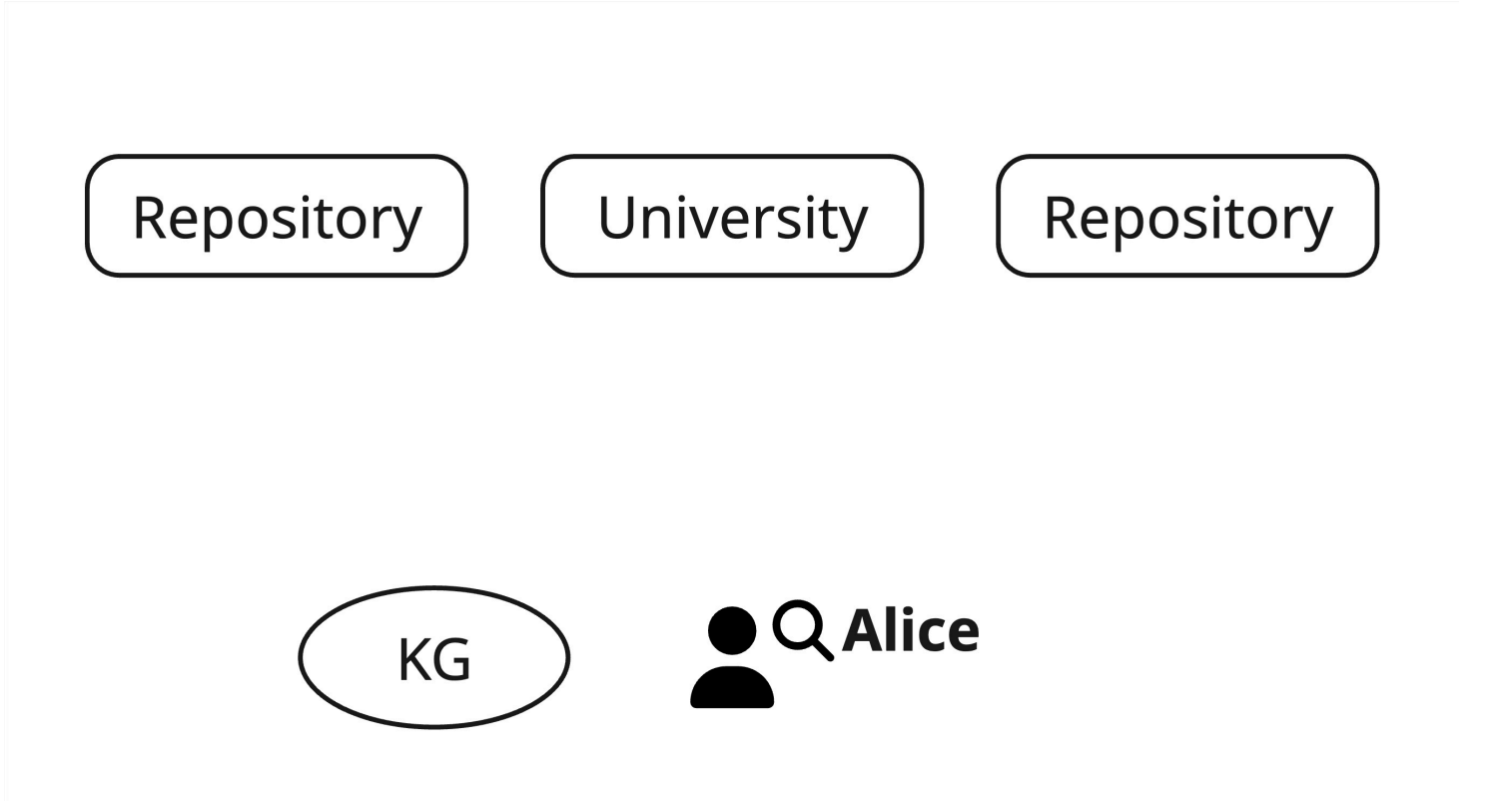
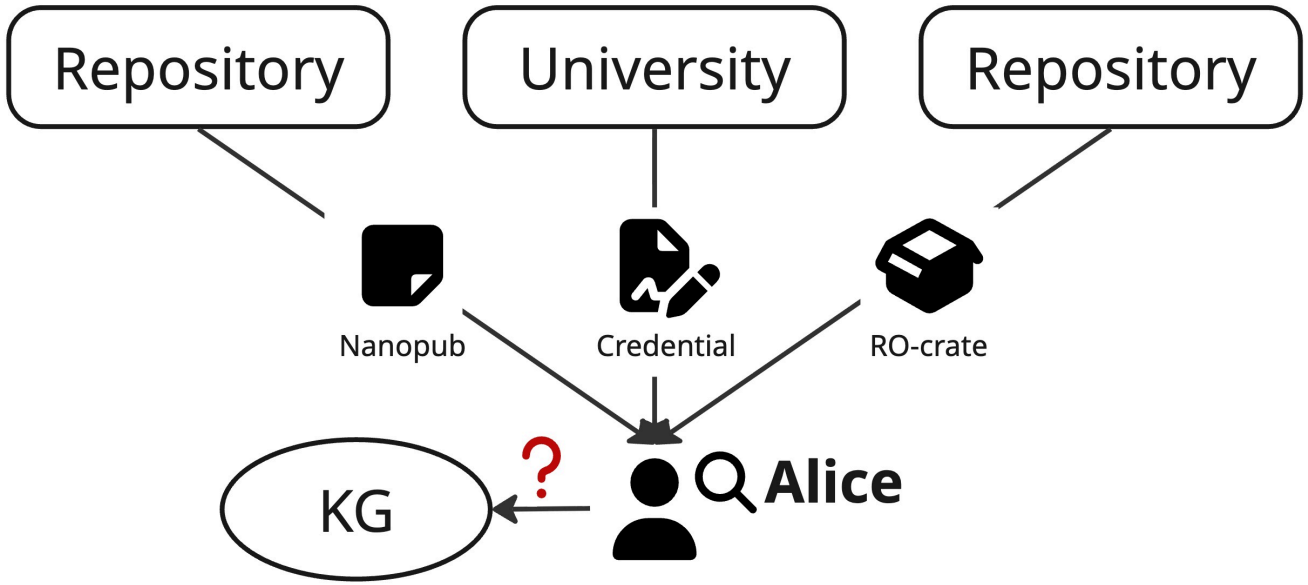
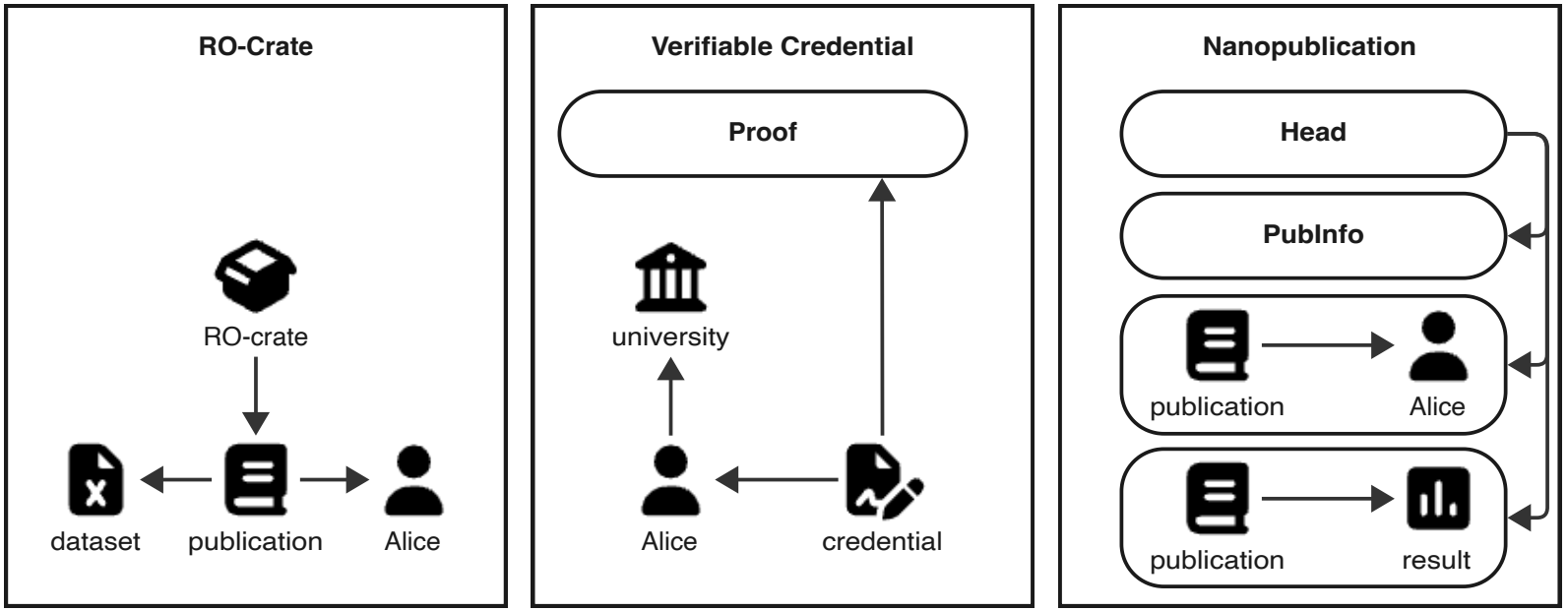
Context Associations: an Application-Independent Annotation Method for RDF Knowledge Graphs

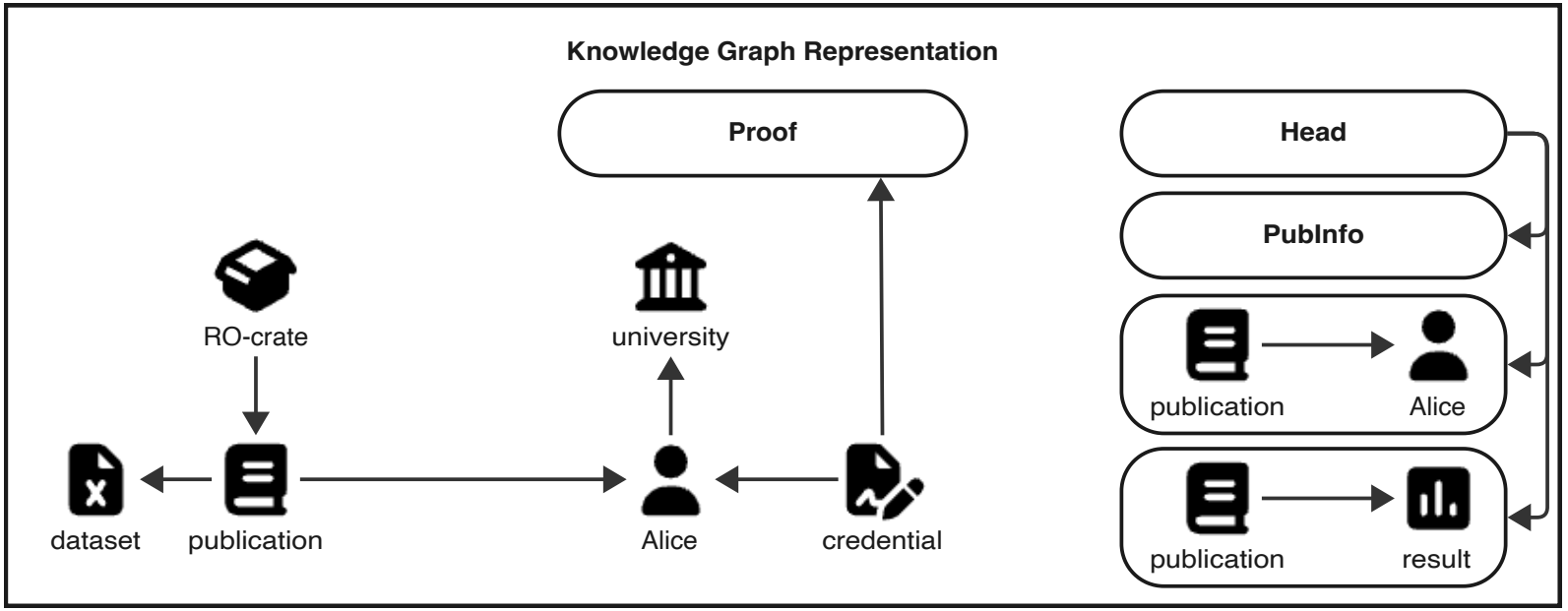
Workshop on Quality of Knowledge Graphs at ESWC 2026

Ruben Dedecker, Ben De Meester, Pieter Colpaert


Repository
University
Repository
KG
Alice


Repository
University
Repository
Nanopub
Credential
RO-crate
KG
?
Alice


RO-Crate
RO-crate
dataset
publication
Alice
Verifiable Credential
Proof
university
Alice
credential
Nanopublication
Head
PubInfo
publication
Alice
publication
result


Knowledge Graph Representation
Proof
Head
PubInfo
RO-crate
university
publication
Alice
dataset
publication
Alice
credential
publication
result

Defining requirements

Context Associations

Querying context

Integrating annotations

Heterogeneous methods

- Hard to store uniformly
- Hard to query generically

Context and data become hard to distinguish

How do we ensure uniform extraction of associated context to target data from KGs

Defining annotation requirements

Interoperable: processed using RDF tooling

Agnostic: can target any set of RDF statements

Explicit: data and context are explicitly defined

Immutable: preserves associations under merge

Recursive: associations can be recursively defined

Defining requirements

Context Associations

Querying context

For a given associated context

```
<Credential> <primarySubject> <Alice>
<Alice> <employee> <University>
<Credential> <proof> <ProofGraph>

<ProofGraph> {
    _:proof <proofValue> "..."
}
```

Graphs for target boundary

```
_:data {
    _:data ca:graphName ca:DefaultGraph
    <Credential> <primarySubject> <Alice>
    <Alice> <employee> <University>
    <Credential> <proof> <ProofGraph>
}

<ProofGraph> {
    _:proof <proofValue> "..."
}
```

Anchor triples to define associations

```
_:data {
    _:data ca:graphName ca:DefaultGraph
    <Credential> <primarySubject> <Alice>
    <Alice> <employee> <University>
    <Credential> <proof> <ProofGraph>
}

<ProofGraph> {
    <ProofGraph> ca:aboutGraph _:data
    _:proof <proofValue> "..."
}
```

Blank nodes to prevent conflicting merges

```
_:data {
    _:data ca:graphName ca:DefaultGraph
    <Credential> <primarySubject> <Alice>
    <Alice> <employee> <University>
    <Credential> <proof> <ProofGraph>
}
_:proofGraph {
    _:proofGraph ca:aboutGraph _:data
                 ca:graphName <ProofGraph>
    _:proof <proofValue> "..."
}
```

Context Associations

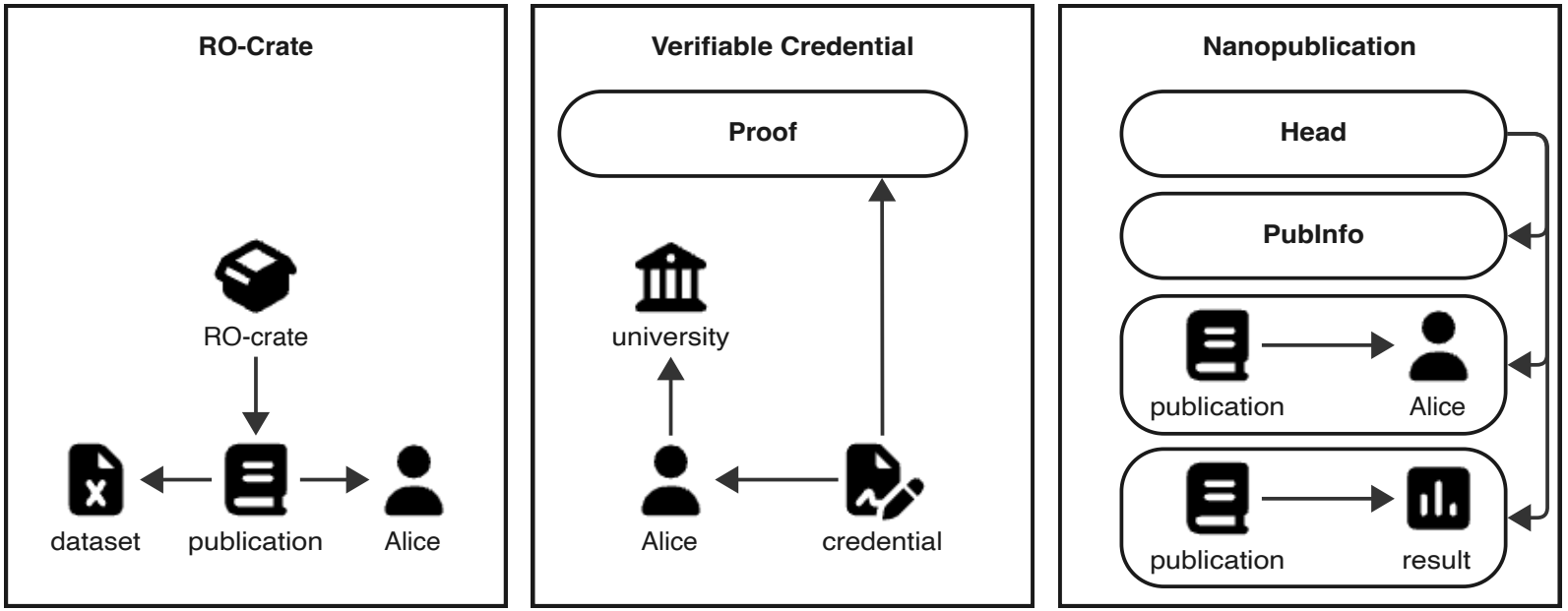
Minimal annotation method using named graphs

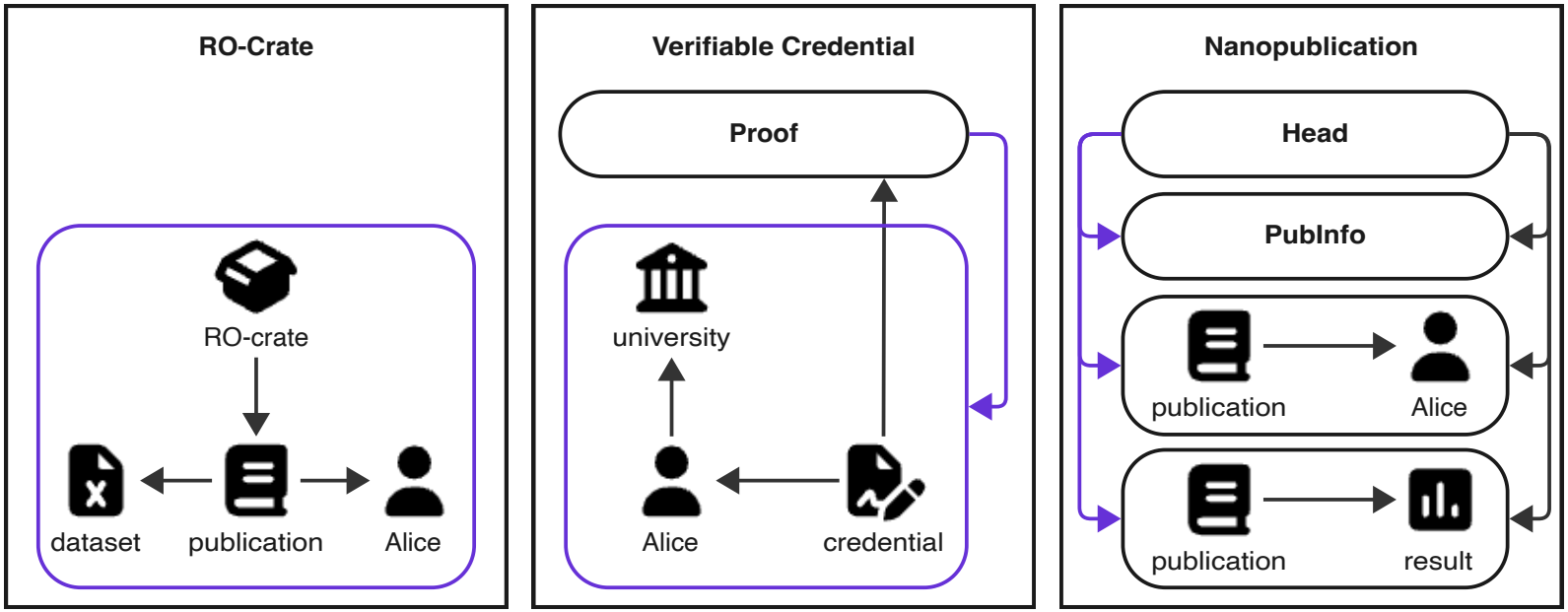
Defines a relation between

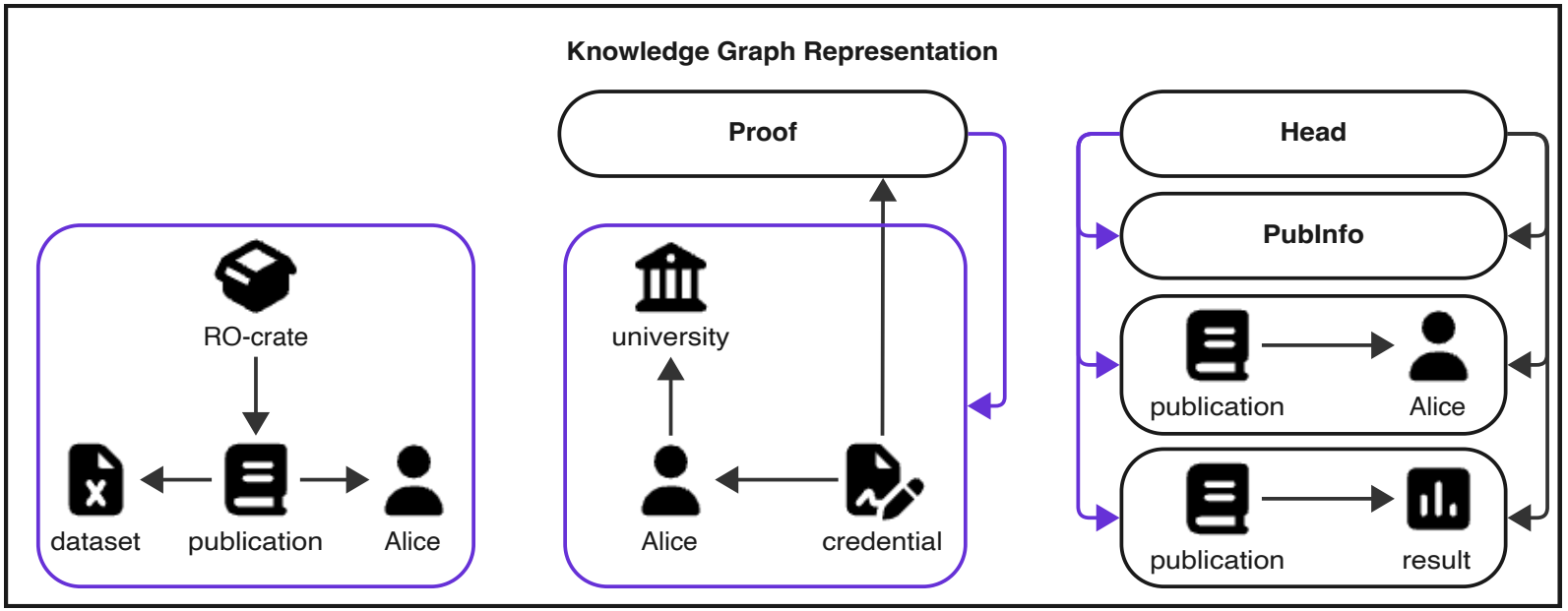
- a named graph of target statements
- a named graph of context statements

through an anchor triple in the context graph

```
_:contextGraph ca:aboutGraph _:targetGraph
```


RO-Crate
RO-crate
dataset
publication
Alice
Verifiable Credential
Proof
university
Alice
credential
Nanopublication
Head
PubInfo
publication
Alice
publication
result

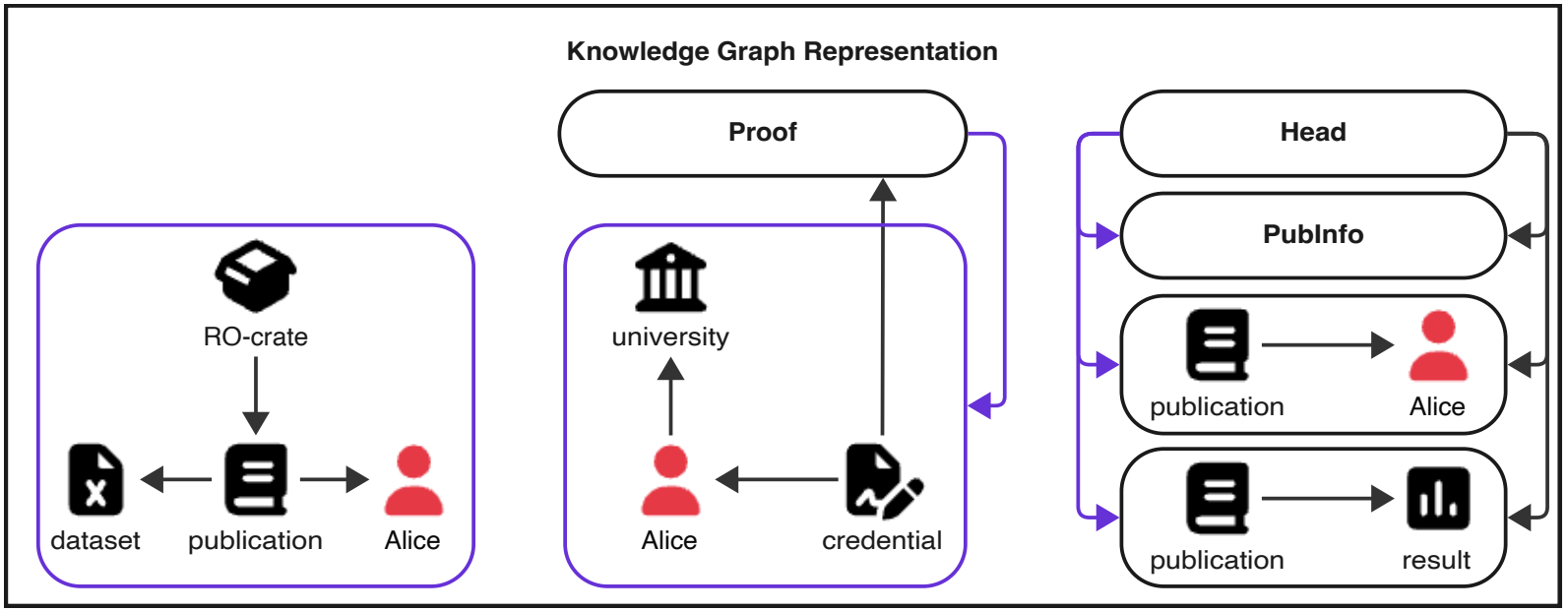

RO-Crate
RO-crate
dataset
publication
Alice
Verifiable Credential
Proof
university
Alice
credential
Nanopublication
Head
PubInfo
publication
Alice
publication
result

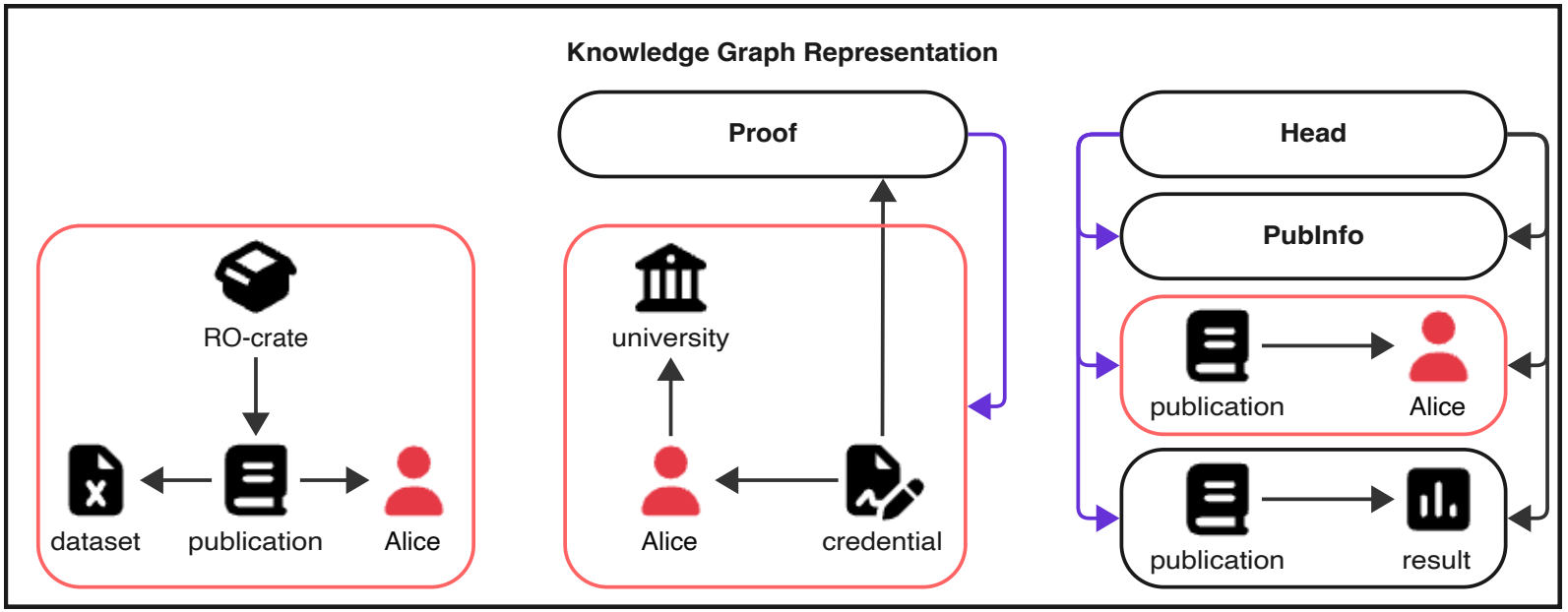

Knowledge Graph Representation
Proof
Head
PubInfo
RO-crate
university
publication
Alice
dataset
publication
Alice
Alice
credential
publication
result

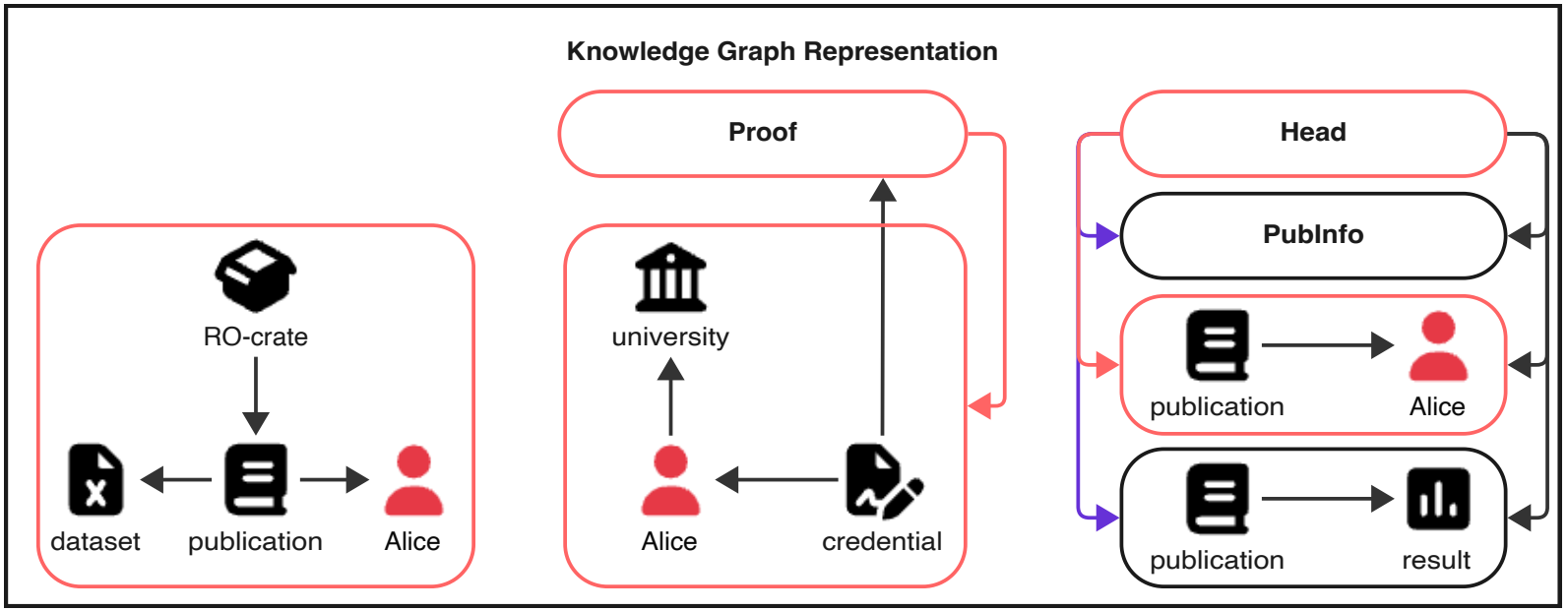
Defining requirements

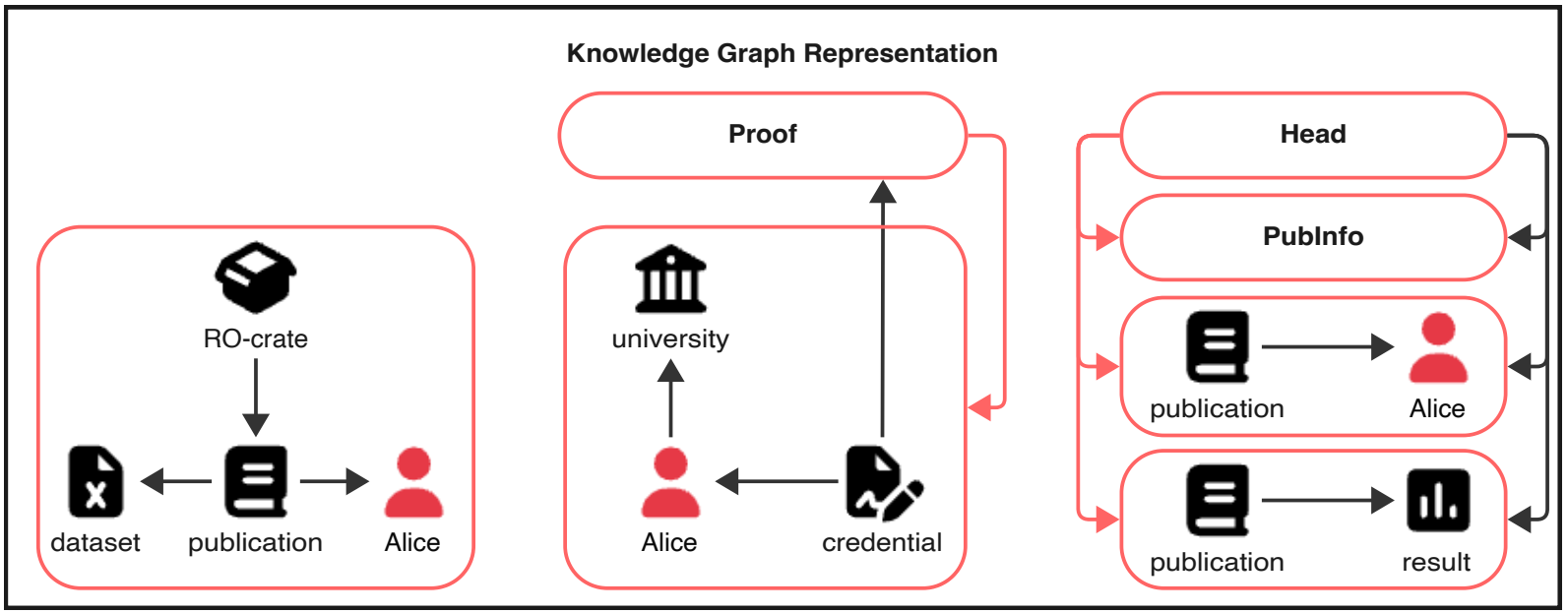
Context Associations

Querying context


Knowledge Graph Representation
Proof
Head
PubInfo
RO-crate
university
publication
Alice
dataset
publication
Alice
Alice
credential
publication
result


Knowledge Graph Representation
Proof
Head
PubInfo
RO-crate
university
publication
Alice
dataset
publication
Alice
Alice
credential
publication
result


Knowledge Graph Representation
Proof
Head
PubInfo
RO-crate
university
publication
Alice
dataset
publication
Alice
Alice
credential
publication
result


Knowledge Graph Representation
Proof
Head
PubInfo
RO-crate
university
publication
Alice
dataset
publication
Alice
Alice
credential
publication
result

Querying with SPARQL

Cannot evaluate variable length property paths over named graphs

Cannot create named graphs with SPARQL construct

Resolved with Apache Jena ARQ

Encoding / decoding

Define organization of graph links

Method-specific requirement (JSON-LD frames)

```
_:g0 {
  <University> <states> _:b0 .
  _:b0 a rdf:Statement .
  _:b0 rdf:subject <Alice> .
  _:b0 rdf:predicate <employee> .
  _:b0 rdf:object <University> .
}
```

```
_:g0 {
  _:g0 ca:aboutGraph _:g1 .
  <University> <states> _:graph .
}
_:g1 {
  _:g1 ca:encodedFrom ca:reification .
  <Alice> <employee> <University> .
}
```

Where does this land us

Interoperable: ~ works with SPARQL, better with ARQ

Agnostic: ✓ complexity in encoding step

Explicit: ✓ explicit data, context and encoding info

Immutable: ✓ stable under dataset merge

Recursive: ✓ requires custom encoding routines

Specification

https://w3id.org/context-associations/specification

ruben.dedecker@ugent.be