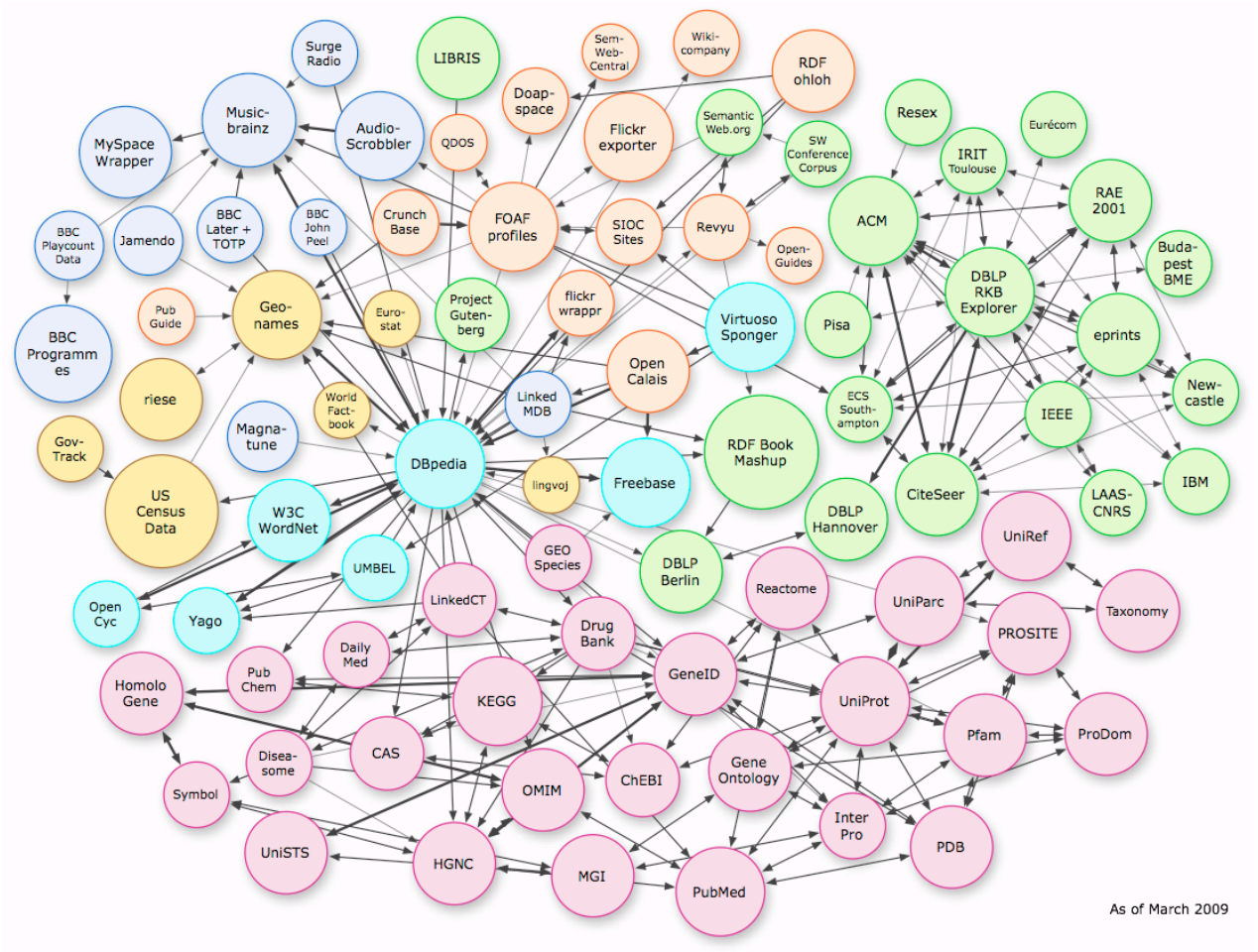
Dynamic provenance for SPARQL Updates using Named Graphs

Harry Halpin (W3C)

James Cheney* (UoE)

* supported by Royal Society University Research Fellowship

“Linked Data”

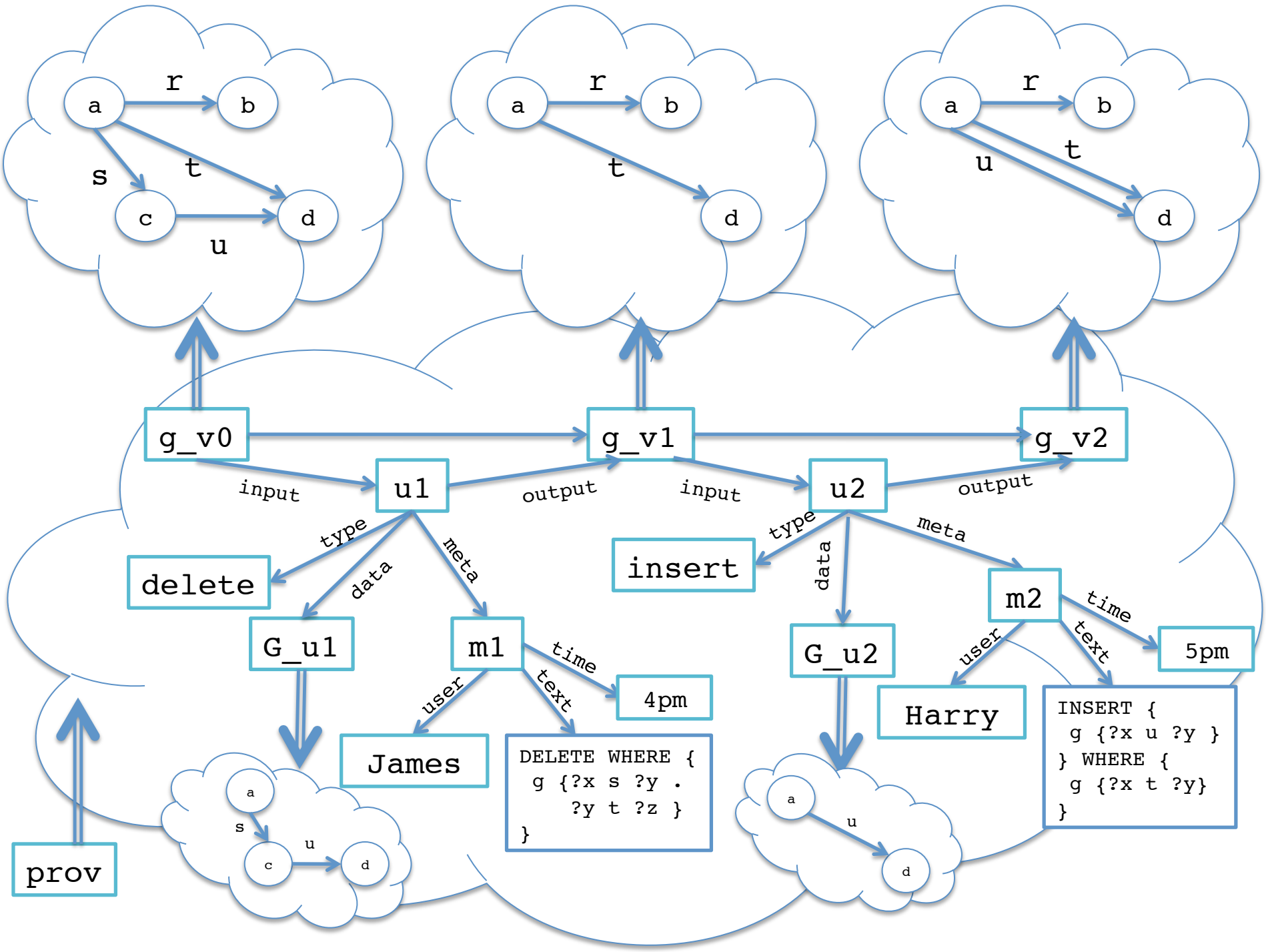

Surge Radio
LIBRIS
Sem-Web-Central
Wiki-company
RDF ohloh
Doap-space
Music-brainz
Audio-Scrobbler
QDOS
Flickr exporter
Semantic Web.org
SW Conference Corpus
Resex
Eurécom
IRIT Toulouse
MySpace Wrapper
BBC Playcount Data
Jamendo
BBC Later + TOTP
BBC John Peel
Crunch Base
FOAF profiles
SIOC Sites
Revyu
Open-Guides
ACM
RAE 2001
Buda-pest BME
DBLP RKB Explorer
Pisa
eprints
Pub Guide
Geo-names
Euro-stat
Project Guten-berg
flickr wrappr
Virtuoso Sponger
BBC Programmes
riese
Open Calais
ECS South-ampton
IEEE
New-castle
World Fact-book
Linked MDB
Magna-tune
Gov-Track
DBpedia
RDF Book Mashup
lingvoj
Freebase
CiteSeer
LAAS-CNRS
IBM
US Census Data
W3C WordNet
DBLP Hannover
UniRef
GEO Species
DBLP Berlin
Reactome
UniParc
Taxonomy
UMBEL
Open Cyc
Yago
LinkedCT
Drug Bank
PROSITE
Daily Med
Pub Chem
GeneID
Homolo Gene
KEGG
UniProt
Pfam
ProDom
Disea-some
CAS
Gene Ontology
Symbol
OMIM
ChEBI
Inter Pro
PDB
UniSTS
HGNC
MGI
PubMed
As of March 2009

Buzzword detox

- **Linked Data** = exporting RDF using stable ids
  - = the part of the "semantic web" that works
- **RDF** = <subject, predicate, object> triples (graphs)
- **Named graphs** = RDF graphs with a (URI) name
  - duh
- **SPARQL** = query language for RDF
  - think conjunctive queries with a few wrinkles
- **SPARQL Updates** = update language for RDF
  - Work in progress (part of SPARQL 1.1)

Dynamic provenance

- By this, I basically just mean version history
  - this would cover large fraction of things people seem to want right off the bat
- Plus minimal ability to track sources of data copied from other places
  - i.e., copy-paste provenance for RDF
- Use of URIs makes this possible.
- This paper: translate updates to self-maintain provenance.


a
r
b
s
t
c
d
u
a
r
b
t
d
a
r
b
u
t
d
g_v0
g_v1
g_v2
input
u1
output
input
u2
output
type
data
meta
delete
G_u1
m1
time
user
text
4pm
James
DELETE WHERE {
g {?x s ?y .
?y t ?z }
}
type
data
meta
insert
G_u2
m2
time
user
text
5pm
Harry
INSERT {
g {?x u ?y }
} WHERE {
g {?x t ?y}
}
prov
a
s
u
c
d
a
u
d

Update language

```
U ::= INSERT {C} WHERE P
    | DELETE {C} WHERE P
    | LOAD g INTO g' | CLEAR g
    | CREATE g | DROP g
```

where C is a SPARQL graph expression (possibly with variables) and P is a SPARQL pattern that queries a graph and binds the variables to URIs or literals.

Graph creation (CREATE g):

```
CREATE g;
CREATE g_v0;
INSERT DATA {GRAPH prov {
   g version g_v0.  g current g_v0.
   u_1 type create.  u_1 output g_v0.
   u_1 meta m_1. ... (other metadata)
   }
}
```

Graph deletion (DROP g):

```
DROP g;
DELETE WHERE {
  GRAPH prov { g current g_vi }
};
INSERT DATA {GRAPH prov {
  u_i type drop.    u_i input g_vi.
  u_i meta m_i.   ... (other metadata)
}
```

Graph load (LOAD g INTO g')

```
LOAD h INTO g;
DELETE WHERE {GRAPH prov {
   g current g_vi
   }
};
INSERT DATA {GRAPH prov {
   g version g_vi+1.  g current g_vi+1.
   u_i type load.  u_i input g_vi.
   u_i output g_vi+1.  u_i source h_j.
   ui meta m_i.  ... (other metadata)
}
```

Insertion (INSERT {C} WHERE P)

```
CREATE g_ui;
INSERT {GRAPH g_ui {C}} WHERE P;
INSERT {GRAPH g {C}} WHERE P;
CREATE g_vi+1;
LOAD g INTO g_vi+1;
DELETE DATA {GRAPH prov {<g current g_vi>}};
INSERT DATA {GRAPH prov {
  g version g_vi+1.        g current g_vi+1.
  u_i input g_vi.          u_i output g_vi+1.
  u_i type insert.         u_i data g_ui.
  u_i source S_1. ...      u_i source S_m.
  u_i meta m_i.    ... (other metadata)
  }
}
```

Also in paper

- Provenance querying via SPARQL
  - implicitly
  - SPARQL is not really enough: no recursion.
  - Not my problem.

- Strawman for provenance retrieval over HTTP

Next steps

- Implementation ☺
  - should be straightforward to implement slow version
- Mapping dynamic provenance from other data models / DBMSs
  - copy-paste DBs/DBWiki
  - export from Oracle with metadata??
- Reconcile with OPM / W3C PIL?