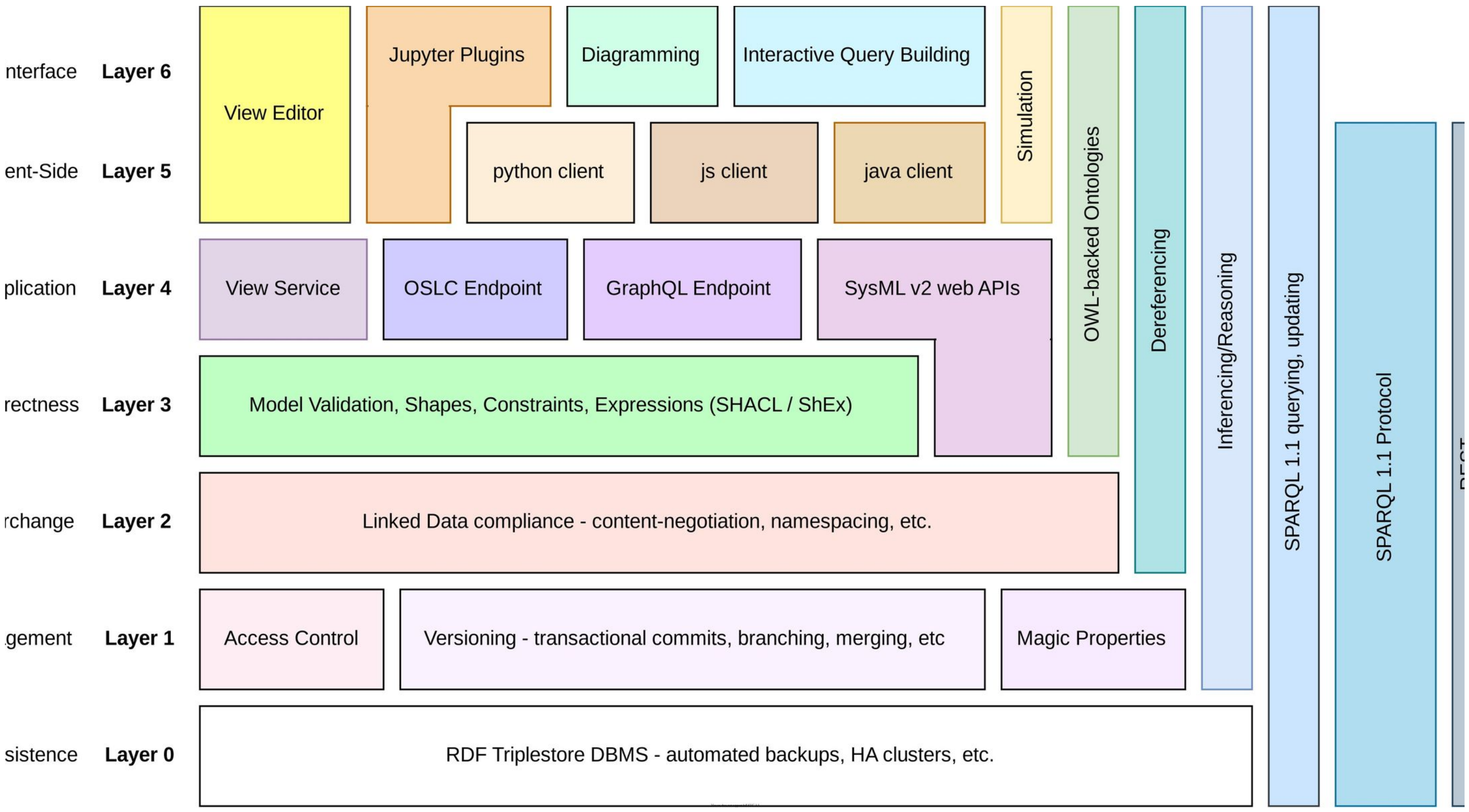
MMS 5 Architecture

Created by chris delp
Last updated: Jun 16, 2021 by Blake Regalia • 4 min read

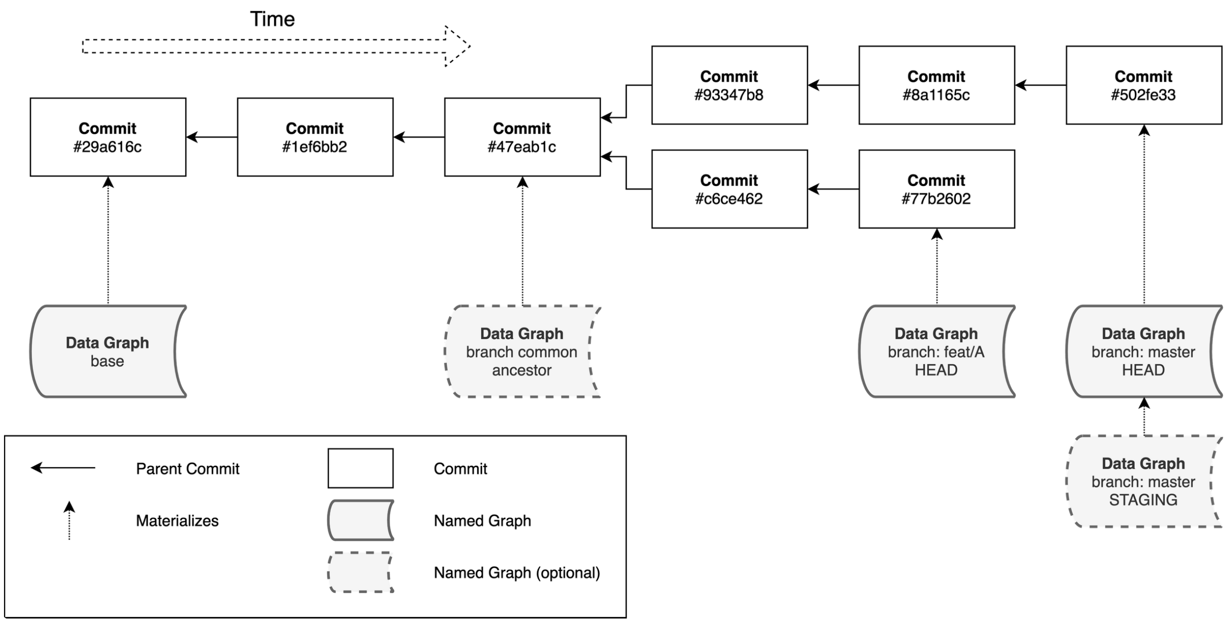
Technology Layer Cake

This diagram locates the involved technologies into or across logical layers that describes how certain components are further abstracted from the underlying RDF triplestore as each task demands more specialized features.

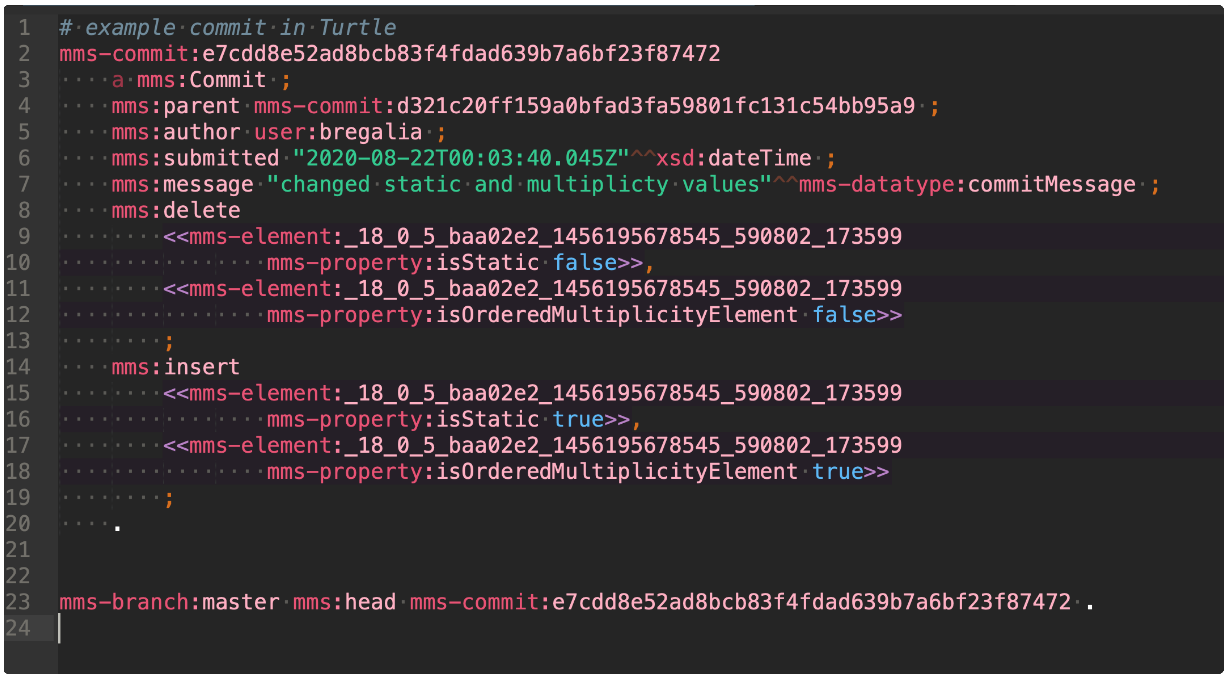


Versioning With Graphs

The proposed versioning mechanism is designed to store consecutive commits as a series of deltas where each commit points to its parent to form a linked tree of commit histories. Additionally, the root commit and the latest HEAD commit are cached such that a copy of the as comparing branches or querying older commits.



To give a concrete example, for those familiar with RDF and Turtle, the following code snippet shows the raw triples for what a hypothetical commit might store to represent a delta in a SysML model.



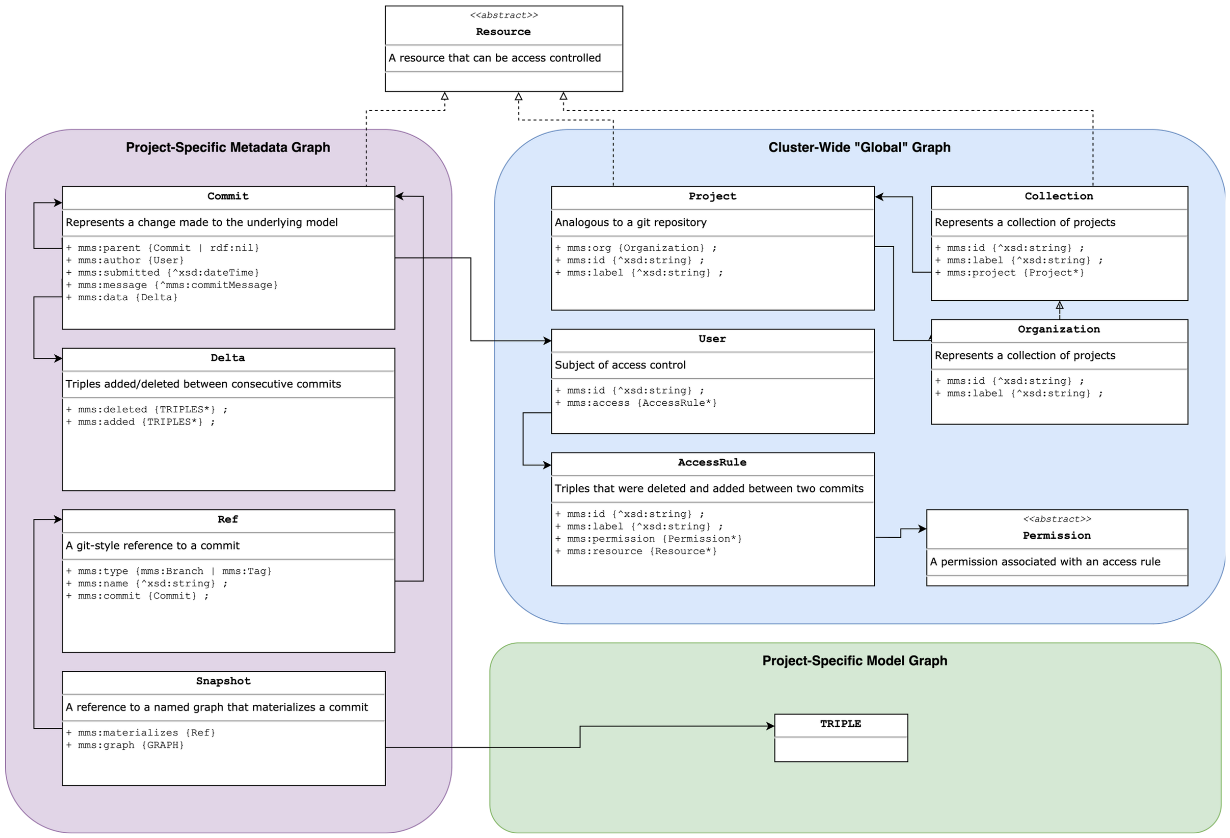
```
# example commit in Turtle
mms-commit:e7cdd8e52ad8bcb83f4fdad639b7a6bf23f87472
    a mms:Commit ;
    mms:parent mms-commit:d321c20ff159a0bfad3fa59801fc131c54bb95a9 ;
    mms:author user:bregalia ;
    mms:submitted "2020-08-22T00:03:40.045Z"^^xsd:dateTime ;
    mms:message "changed static and multiplicty values"^^mms-datatype:commitMessage ;
    mms:delete
        <<mms-element:_18_0_5_baa02e2_1456195678545_590802_173599
                mms-property:isStatic false>>,
        <<mms-element:_18_0_5_baa02e2_1456195678545_590802_173599
                mms-property:isOrderedMultiplicityElement false>>
        ;
    mms:insert
        <<mms-element:_18_0_5_baa02e2_1456195678545_590802_173599
                mms-property:isStatic true>>,
        <<mms-element:_18_0_5_baa02e2_1456195678545_590802_173599
                mms-property:isOrderedMultiplicityElement true>>
        ;
    .


mms-branch:master mms:head mms-commit:e7cdd8e52ad8bcb83f4fdad639b7a6bf23f87472 .

```

Model Management with Named Graphs

organization they belong to, the users, access rules, and so on. This named graph has a known IRI and provides an entry-point for project discovery. Each project has its own Metadata graph (shown in lavender) which stores all the versioning information such as commit history, deltas, and refs. Finally, a Snapshot object stored in the Metadata graph associates a Ref to a materialized Model graph (shown in green) which stores exclusively the entire model data for a given commit. Most user queries will likely focus on the Model graph, while some use cases will also involve the Metadata graph (for example building time-series diagrams or querying the commit history for certain changes).

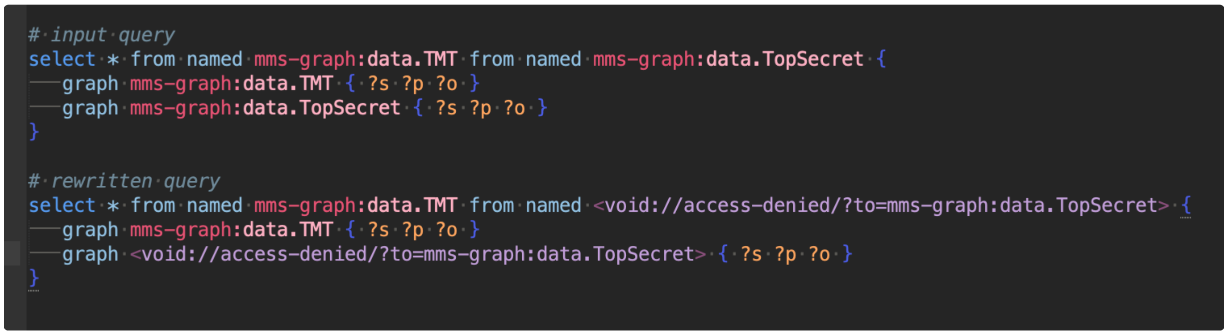

<<abstract>>
Resource
A resource that can be access controlled
Project-Specific Metadata Graph
Commit
Represents a change made to the underlying model
+ mms:parent {Commit | rdf:nil}
+ mms:author {User}
+ mms:submitted {^xsd:dateTime}
+ mms:message {^mms:commitMessage}
+ mms:data {Delta}
Delta
Triples added/deleted between consecutive commits
+ mms:deleted {TRIPLES*} ;
+ mms:added {TRIPLES*} ;
Ref
A git-style reference to a commit
+ mms:type {mms:Branch | mms:Tag}
+ mms:name {^xsd:string} ;
+ mms:commit {Commit} ;
Snapshot
A reference to a named graph that materializes a commit
+ mms:materializes {Ref}
+ mms:graph {GRAPH}
Cluster-Wide "Global" Graph
Project
Analogous to a git repository
+ mms:org {Organization} ;
+ mms:id {^xsd:string} ;
+ mms:label {^xsd:string} ;
Collection
Represents a collection of projects
+ mms:id {^xsd:string} ;
+ mms:label {^xsd:string} ;
+ mms:project {Project*}
User
Subject of access control
+ mms:id {^xsd:string} ;
+ mms:access {AccessRule*}
Organization
Represents a collection of projects
+ mms:id {^xsd:string} ;
+ mms:label {^xsd:string} ;
AccessRule
Triples that were deleted and added between two commits
+ mms:id {^xsd:string} ;
+ mms:label {^xsd:string} ;
+ mms:permission {Permission*}
+ mms:resource {Resource*}
<<abstract>>
Permission
A permission associated with an access rule
Project-Specific Model Graph
TRIPLE

Access Control

For multi-tenant installations, MMS must ensure that users are only able to query data from the projects they are authorized to access. This is accomplished via SPARQL query rewriting for ASK, SELECT and DESCRIBE queries. At the moment, MMS does not yet plan to support arbitrary UPDATE queries from users, although the same mechanism would still apply.

The process for restricting access to certain project data via SPARQL query rewriting is simple. The SPARQL query is parsed, and a set of rules is applied on the resulting abstract syntax tree.

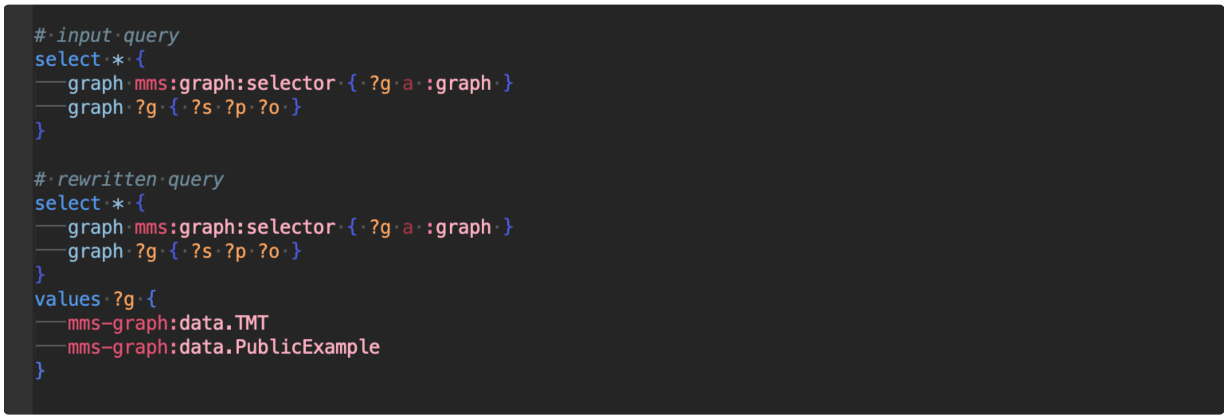
1. When using `FROM NAMED` with IRIs to query specific graphs, named graphs that the user does not have access to are rewritten as `<void://access-denied/?to=$GRAPH>` :



```
# input query
select * from named mms-graph:data.TMT from named mms-graph:data.TopSecret {
    graph mms-graph:data.TMT { ?s ?p ?o }
    graph mms-graph:data.TopSecret { ?s ?p ?o }
}

# rewritten query
select * from named mms-graph:data.TMT from named <void://access-denied/?to=mms-graph:data.TopSecret> {
    graph mms-graph:data.TMT { ?s ?p ?o }
    graph <void://access-denied/?to=mms-graph:data.TopSecret> { ?s ?p ?o }
}
```

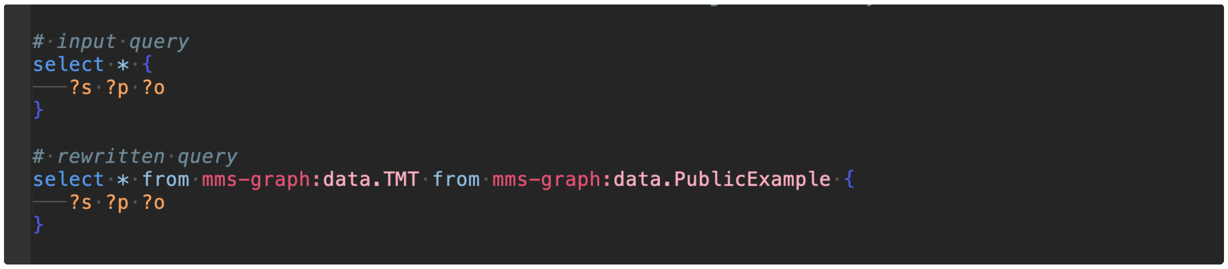
2. When using `GRAPH` with variables to query an arbitrary graph, a `VALUES` clause is added to restrict the bindings that the graph variable can take:



```
# input query
select * {
    graph mms:graph:selector { ?g a :graph }
    graph ?g { ?s ?p ?o }
}

# rewritten query
select * {
    graph mms:graph:selector { ?g a :graph }
    graph ?g { ?s ?p ?o }
}
values ?g {
    mms-graph:data.TMT
    mms-graph:data.PublicExample
}
```

3. When using the default graph to perform an implicit union, a list of graphs the user can access is added to the select clause using the `FROM` keywords:



```
# input query
select * {
    ?s ?p ?o
}

# rewritten query
select * from mms-graph:data.TMT from mms-graph:data.PublicExample {
    ?s ?p ?o
}
```

Any combination of the above operations will still work and the rewriting process is not order-dependent. The service only needs to know about the finite set of projects the current user has access to.