Browsing RDF – take 4

Dr. Ora Lassila

Founder, So Many Aircraft

(also: Principal Technologist, Amazon Neptune)

2024-12-06 (107th anniversary of Finland's independence)

Introduction

Current:

Principal Technologist, Amazon Neptune

Co-chair, W3C RDF-star WG

Past:

State Street, Pegasystems, Nokia, MIT, CMU, Helsinki Univ. of Tech.

Education:

Ph.D (D.Sc) CS & AI, Helsinki Univ. of Tech.

So Many Aircraft:

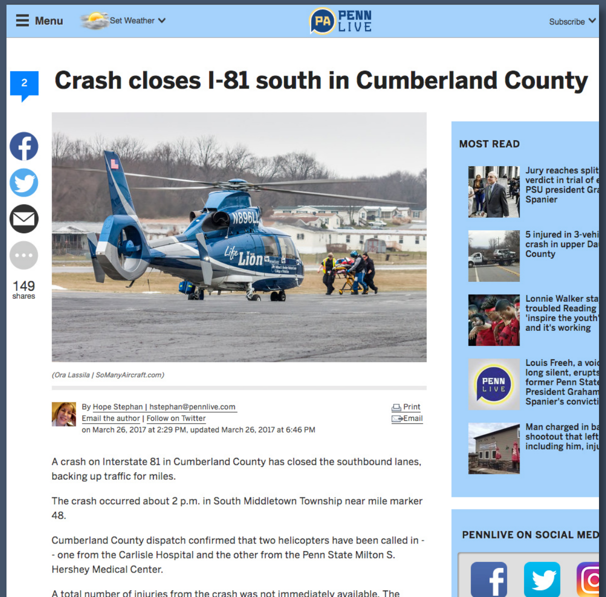
My "side hustle", started in earnest in 2014

Published two books (so far)

Maintaining an aviation reference library (4,500+ books and magazines) and a large photo collection (70,000+ images)

Knowledge graph capturing all of the above

AS.365 Dauphin II
in Emergency Medical Service
Ora Lassila
Reference Guide for Modelers and Helicopter Enthusiasts
So Many Aircraft – volume 3



Crash closes I-81 south in Cumberland County

(Ora Lassila | SoManyAircraft.com)

By Hope Stephan | hstephan@pennlive.com
Email the author | Follow on Twitter
on March 26, 2017 at 2:29 PM, updated March 26, 2017 at 6:46 PM

A crash on Interstate 81 in Cumberland County has closed the southbound lanes, backing up traffic for miles.

The crash occurred about 2 p.m. in South Middletown Township near mile marker 48.

Cumberland County dispatch confirmed that two helicopters have been called in -- one from the Carlisle Hospital and the other from the Penn State Milton S. Hershey Medical Center.

A total number of injuries from the crash was not immediately available. The

Jury reaches split verdict in trial of e PSU president Gra Spanier

5 injured in 3-vehi crash in upper Da County

Lonnie Walker sta troubled Reading 'inspire the youth' and it's working

Louis Freeh, a voic long silent, erupts former Penn State President Graham Spanier's convicti

Man charged in ba shootout that left including him, inju

Me and KR platforms…

My M.Sc thesis, KR layer for many projects

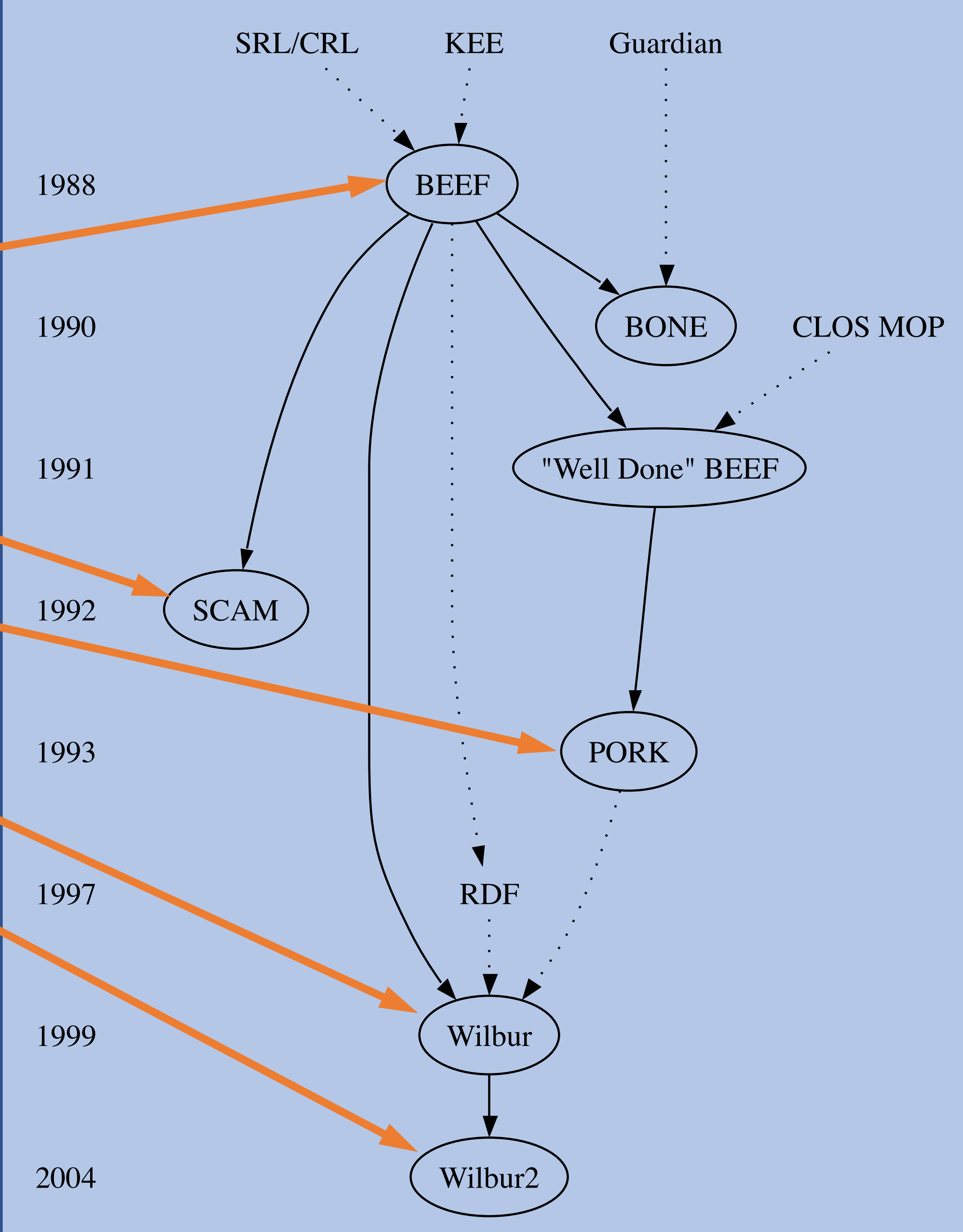
KR layer for NASA's RAX planner

KR layer for CMU's OZONE platform

My 1st attempt to build an RDF library

My Ph.D thesis (path queries, reasoning)

Piglet: Wilbur redone in C++
and Python (in 2008, not shown)



From [Lassila 2007]

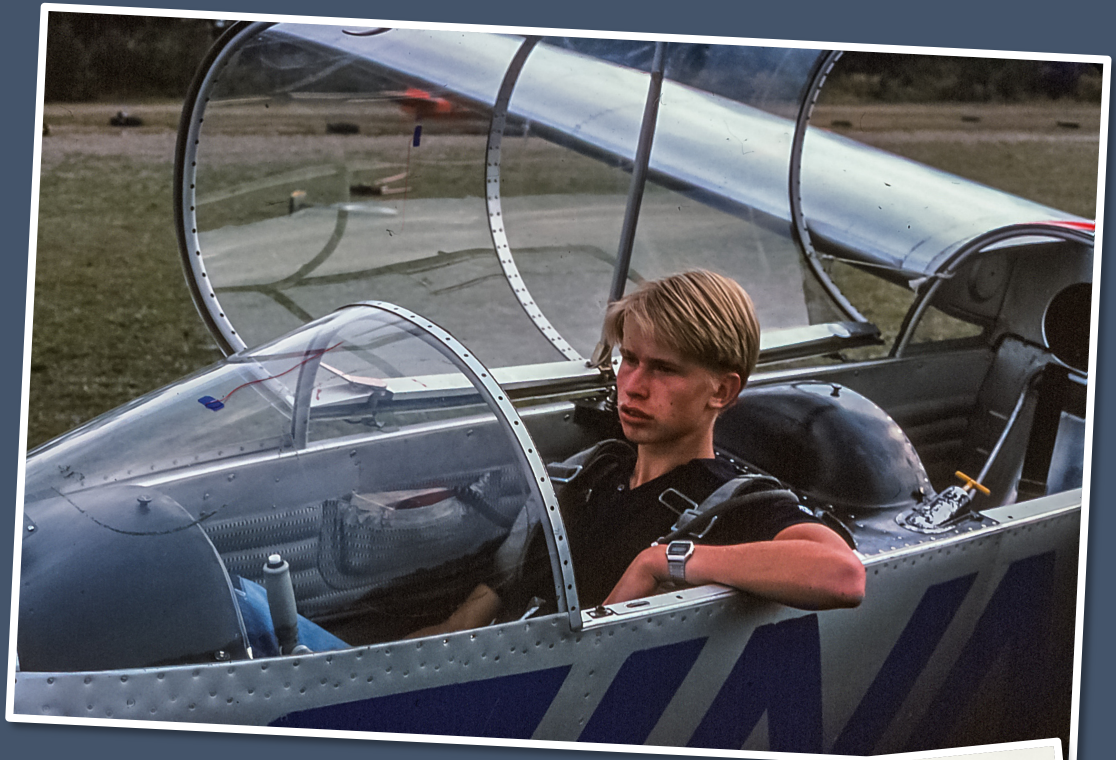
Me and aviation…

Started taking aircraft photos at age 12.

First solo flight at age 17 (glider certificate).

Wanted to be a fighter pilot (instead, served in the Finnish Army).

Life-long builder of scale models.



Browsing RDF?

Browsing RDF is effectively step-by-step path traversal

We define a *traversal query* as a means of discovering what is immediately next to the node we are currently looking at

- in other words, “where could the user go next?”
- this could mean “forwards” or “backwards”

Traversal queries cannot terminate in a blank node, because subsequent queries cannot start from a blank node

- we need to be able to do this in a stateless system (approx. REST)

Browsing RDF?

RDF Concise Bounded Description (CBD) is a recursive traversal query guaranteed not to terminate in a blank node

Most triple store implementations return a CBD for DESCRIBE queries

CBD cannot be expressed as a SELECT query

- path-length -bounded approximations can be written as SELECT queries

https://www.w3.org/submissions/CBD/

Browsing RDF?

A browsable graph can be rendered one CBD at a time

The RDF metamodel and RDF Schema are all the system needs to know to render RDF in a human-friendly manner

- any ontologies the system has access to just make the experience better
- "best effort"

Browsing RDF?

Basic principles:

1. every node becomes a page in HTML hypertext, every (owl:ObjectProperty) edge becomes a link to another page
2. never show a URI if a qualified name can be derived, never show a qualified name if a human-readable label can be found; use rdfs:label (and sub-properties thereof) to find labels
3. everything is "clickable"
4. provide means to filter what is shown (e.g., literal language tags)
5. provide means to customize rendering (e.g., typed literals)

My attempts to build an “RDF browser”

Wilbur	2000	MCL IDE + CLOS	direct API	Extension of MCL’s data inspector. Wilbur toolkit open sourced for the DAML program.
OINK 1	2005	Wilbur + CLOS	WQL	Part of my Ph.D work to demonstrate WQL. Included “query-by-browsing” UI.
OINK 2	2008	Piglet + C++/C + Python 2.7	WQL	Trying to put a triple store on a mobile phone. Derailed by Nokia’s downfall.
OINK 3	2022	Any triple store + Python 3.x	SPARQL	Trying to see if this can be done with SPARQL alone. Managing SMA’s KG.

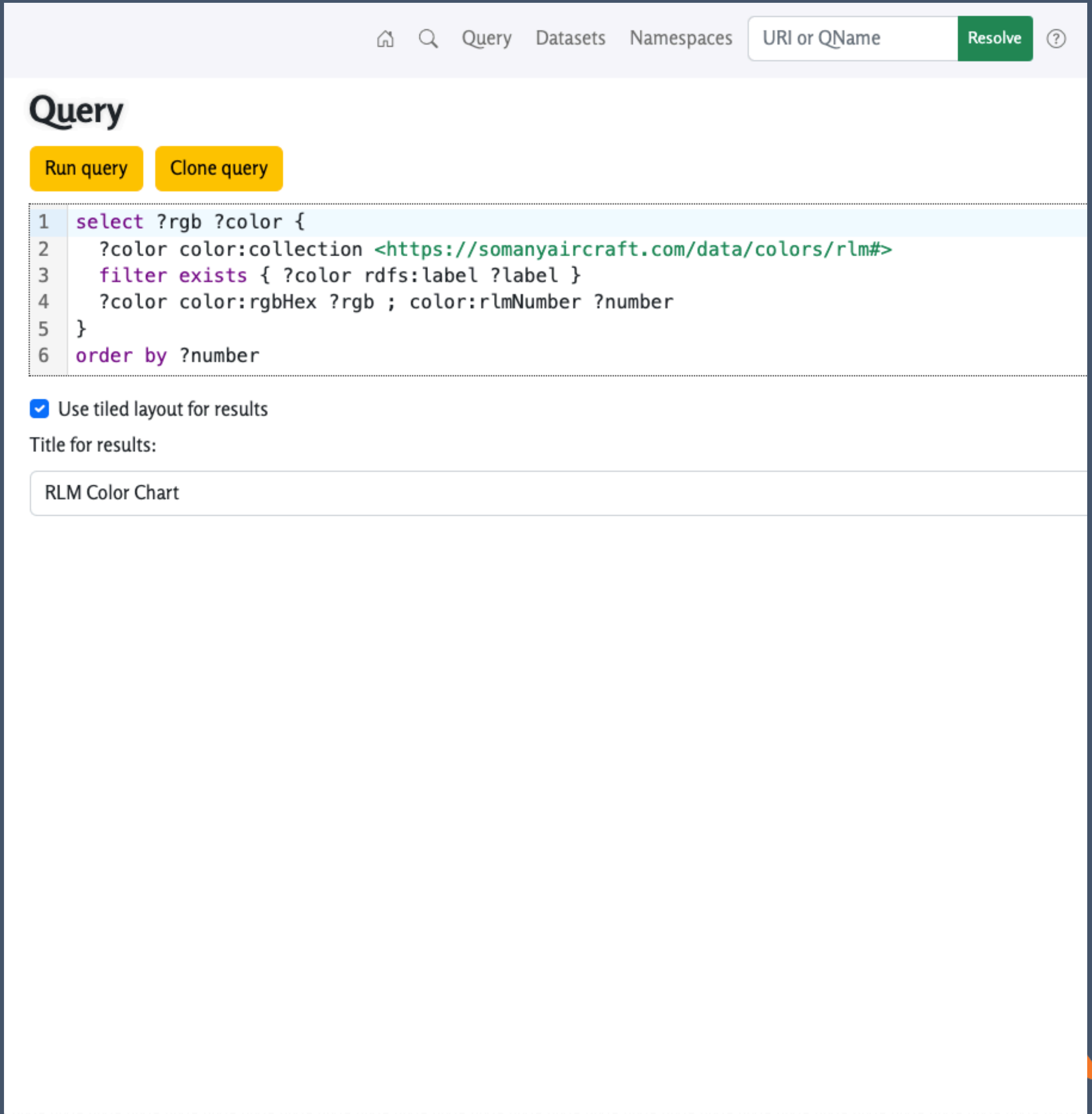
Query editor

Queries can be edited, visualized, and saved

"Saving" a query simply means you bookmark it

Query results are "browsable"

Demo later…



Query

```
select ?rgb ?color {
  ?color color:collection <https://somanyaircraft.com/data/colors/rlm#>
  filter exists { ?color rdfs:label ?label }
  ?color color:rgbHex ?rgb ; color:rlmNumber ?number
}
order by ?number
```

☑ Use tiled layout for results

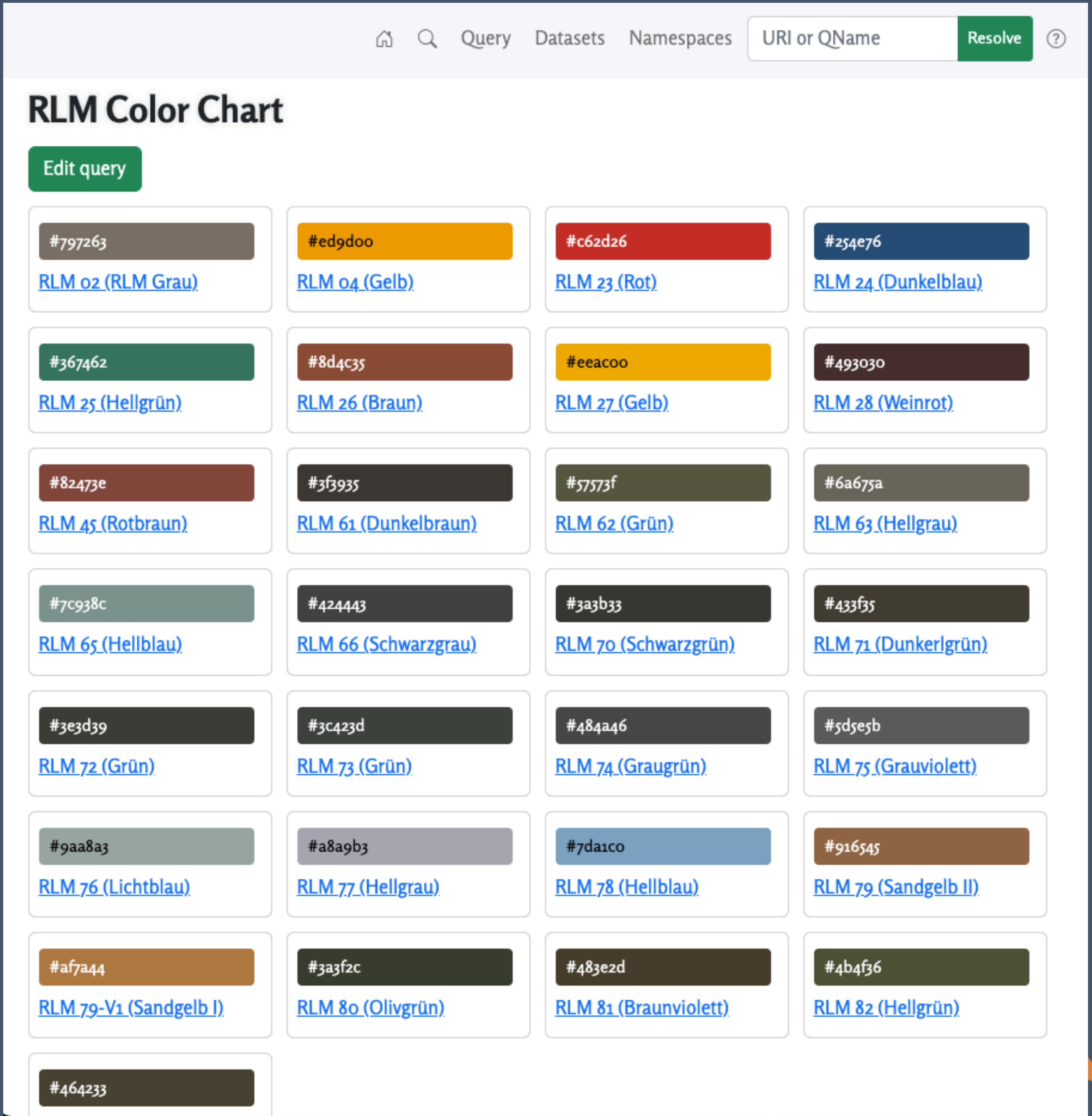
Title for results: RLM Color Chart

Which one is better?

```
<https://somanyaircraft.com/data/colors/rlm#02> a color:RLMColor ;
  rdfs:label "RLM 02 (RLM Grau)" ;
  color:collection <https://somanyaircraft.com/data/colors/rlm#> ;
  color:name "RLM Grau" ;
  color:rgbHex "797263"^^color:RGB ;
  color:rlmNumber "02" ;
  color:substitute [ a color:RLMColor ;
      dcterms:source <https://hyperscale.com/images/Digital%20RLM%20Colours.pdf> ;
      color:name "RLM Grau" ;
      color:rgbHex "7f7b6b"^^color:RGB ;
      color:rlmNumber "02" ] .

<https://somanyaircraft.com/data/colors/rlm#04> a color:RLMColor ;
  rdfs:label "RLM 04 (Gelb)" ;
  color:collection <https://somanyaircraft.com/data/colors/rlm#> ;
  color:name "Gelb" ;
  color:rgbHex "ed9d00"^^color:RGB ;
  color:rlmNumber "04" ;
  color:substitute [ a color:RLMColor ;
      dcterms:source <https://hyperscale.com/images/Digital%20RLM%20Colours.pdf> ;
      color:name "Gelb" ;
      color:rgbHex "ef9d00"^^color:RGB ;
      color:rlmNumber "04" ],
    <https://somanyaircraft.com/data/colors/ral#1003> .

<https://somanyaircraft.com/data/colors/rlm#23> a color:RLMColor ;
  rdfs:label "RLM 23 (Rot)" ;
  color:collection <https://somanyaircraft.com/data/colors/rlm#> ;
  color:name "Rot" ;
  color:rgbHex "c62d26"^^color:RGB ;
  color:rlmNumber "23" ;
  color:substitute [ a color:RLMColor ;
```

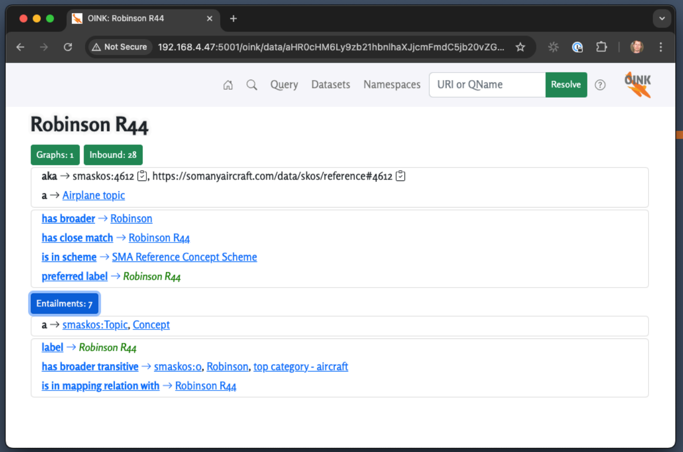


RLM Color Chart

Edit query

#797263 RLM 02 (RLM Grau)	#ed9d00 RLM 04 (Gelb)	#c62d26 RLM 23 (Rot)	#254e76 RLM 24 (Dunkelblau)
#367462 RLM 25 (Hellgrün)	#8d4c35 RLM 26 (Braun)	#eeac00 RLM 27 (Gelb)	#493030 RLM 28 (Weinrot)
#82473e RLM 45 (Rotbraun)	#3f3935 RLM 61 (Dunkelbraun)	#57573f RLM 62 (Grün)	#6a675a RLM 63 (Hellgrau)
#7c938c RLM 65 (Hellblau)	#424443 RLM 66 (Schwarzgrau)	#3a3b33 RLM 70 (Schwarzgrün)	#433f35 RLM 71 (Dunkerlgrün)
#3e3d39 RLM 72 (Grün)	#3c423d RLM 73 (Grün)	#484a46 RLM 74 (Graugrün)	#5d5e5b RLM 75 (Grauviolett)
#9aa8a3 RLM 76 (Lichtblau)	#a8a9b3 RLM 77 (Hellgrau)	#7da1c0 RLM 78 (Hellblau)	#916545 RLM 79 (Sandgelb II)
#af7a44 RLM 79-V1 (Sandgelb I)	#3a3f2c RLM 80 (Olivgrün)	#483e2d RLM 81 (Braunviolett)	#4b4f36 RLM 82 (Hellgrün)
#464233			

OINK architecture



Robinson R44

Graphs: 1 Inbound: 28

aka → smaskos:4612, https://somanyaircraft.com/data/skos/reference#4612

a → Airplane topic

has broader → Robinson

has close match → Robinson R44

is in scheme → SMA Reference Concept Scheme

preferred label → *Robinson R44*

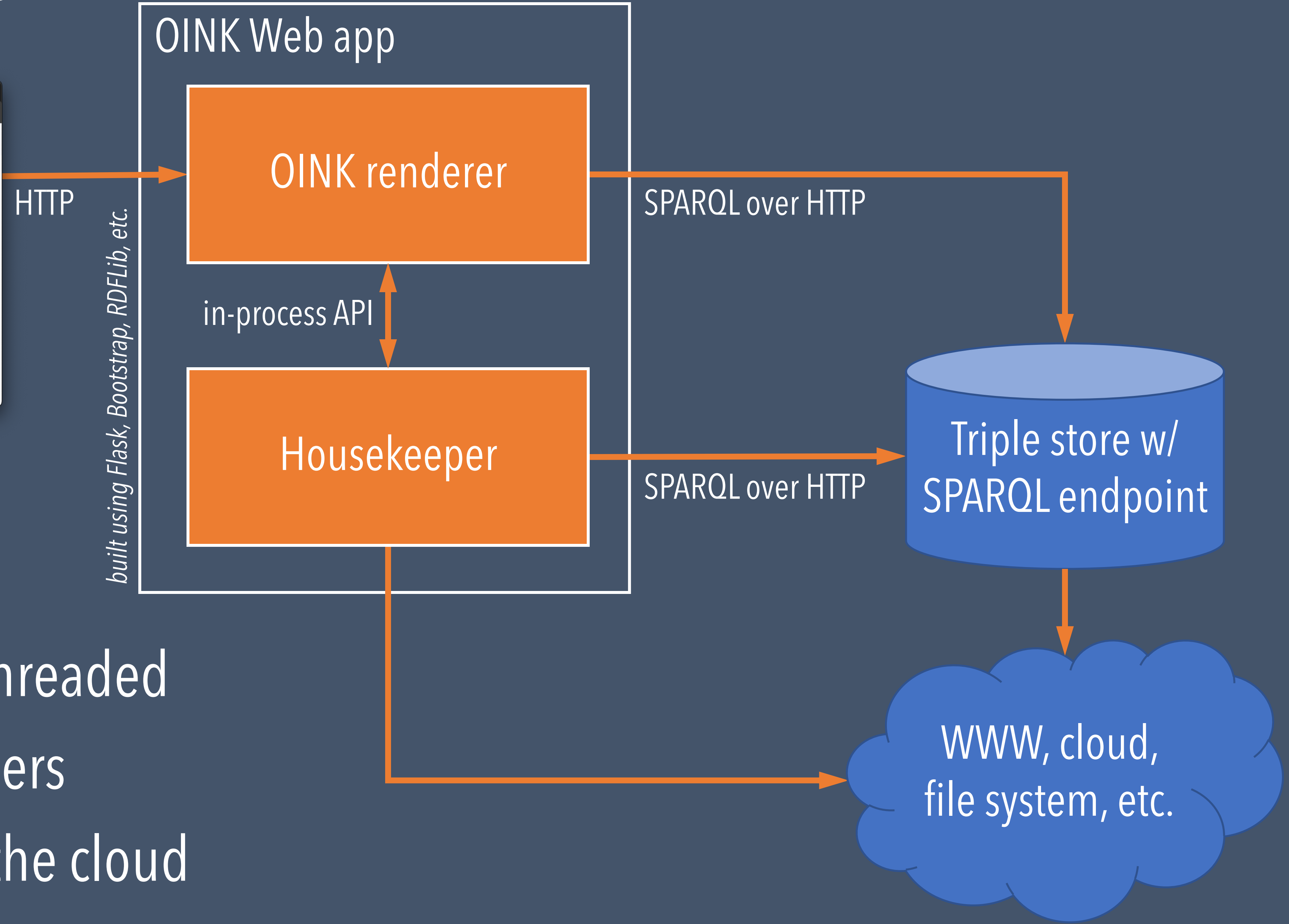
Entailments: 7

a → smaskos:Topic, Concept

label → *Robinson R44*

has broader transitive → smaskos:0, Robinson, top category - aircraft

is in mapping relation with → Robinson R44



HTTP

built using Flask, Bootstrap, RDFLib, etc.

OINK Web app

OINK renderer

in-process API

Housekeeper

SPARQL over HTTP

SPARQL over HTTP

Triple store w/ SPARQL endpoint

WWW, cloud, file system, etc.

OINK Web app

- is stateless & multi-threaded
- supports multiple users
- can run locally or in the cloud

Housekeeper

Maintains a catalog of datasets in the triple store (using DCAT)

- tracks catalog changes (using PROV-O)
- supports classification of data sources and datasets (using SKOS)

Can load any serialized RDF via HTTP or from the local file system

Can load RDF from tabular or semi-structured sources (using RML)

- we have our own implementation of RML ("tinyrml" Python lib)

Can read image metadata directly from photos (XMP)

- we have our own XMP support ("xmptools" Python lib)

Ongoing development

Current SMA KG lives in a system built on the Wilbur RDF toolkit

- Wilbur triple store → no SPARQL
- UI is not flexible enough to handle the full diversity of the KG
- photo metadata is not automatically harvested

Upcoming additions to OINK

- UI layer with bespoke class-based rendering; current OINK UI as fallback
- SPARQL update queries

SMA KG migrated to a separate triple store, new UI built on OINK

Demo

Questions?

ora@somanyaircraft.com
ora@amazon.com

https://www.somanyaircraft.com/