Getty Vocabularies: Linked Open Data

Semantic Representation

Version: 2.0
Last updated: 15 Jun 2014
HTML version: http://vocab.getty.edu/doc/ (for linking)
PDF version: http://vocab.getty.edu/doc/gvp-lod.pdf (for printing)
Formerly at: http://www.getty.edu/research/tools/vocabularies/lod/aat_semantic_representation.pdf

Table of Contents

1 Introduction

This document explains the representation of the Art and Architecture Thesaurus (AAT)® in semantic format, using RDF and appropriate ontologies. It is published in PDF, together with the Illustrations in an accompanying ZIP file. We hope to be able soon to publish it in HTML.

1.1 The Getty Vocabularies and LOD

The Getty Vocabularies were first built to help people categorize, describe, and index cultural heritage objects and information. Now we have the technology to transform these human-defined relationships into machine-readable data sets and embed them into the evolving semantic web. By publishing the Getty Vocabularies as Linked Data in an open environment for anyone to freely use, we are sharing with the world the results of over thirty years of research and scholarship.

1.1.1 About the AAT

The AAT is a structured, multilingual vocabulary including terms, descriptions, and other information for generic concepts related to art, architecture, other cultural heritage, and conservation. For decades now, the AAT has been used as a primary reference by museums, art libraries, archives, visual resource catalogers, conservation specialists, archaeological projects, bibliographic projects, researchers, and information specialists who are dealing with the needs of these users. Like all of the Getty Vocabularies, the AAT is compliant with international standards and grows through contribution.

The AAT includes info about 42k Subjects, 300k Terms, 84k Scope Notes, 40k Sources and 165 Contributors (see Counting).

For more information about the history, purpose and scope of the AAT see:
http://www.getty.edu/research/tools/vocabularies/aat/about.html

1.2 Revisions, Review, Feedback

Initial version: Vladimir Alexiev, Joan Cobb, Gregg Garcia, Patricia Harpring.

Updates: Vladimir Alexiev

1.2.1 Revisions

1.2.1.1 Version 1.0

19 Feb 2014: Initial Version

1.2.1.2 Version 1.1

28 Feb 2014: Draft that was posted for about 2 weeks. Do not use.

1.2.1.3 Version 1.2

14 Mar 2014

- Mapping change: removed intermediate bibo:DocumentPart node in Language Tags and Sources
- Mapping change: removed Top Concepts indication since it is not considered useful
- Mapping change: touched Relationship Representation a bit to fit better the used ontology documentation tool (Parrot).
- Added comprehensive Descriptive Information (VOID etc)
- Added documentation: Language Tag Case, ontology documentation at end of GVP Ontology, clarification at end of Hierarchy Structure, clarification at end of Language Dual URLs.
- Sample Queries: added language-related (Find Subject by Exact English PrefLabel, Find Subject by Language-Independent PrefLabelsFind Subject by Any Label, Find Terms by Language Tag, Languages and ISO Codes, Language URLs), added Explore the Ontology (3 queries), updated the queries in the SPARQL endpoint, provided link to this documentation (exclamation point icon)
- Site change: removed parasitic word "/resource" from GVP URLs

1.2.1.4 Version 1.3

15 Apr 2014

- Mapping change: omitted Sort Order of Facets and Hierarchies since it's not considered useful
- Omitted blank nodes from Per-Entity Exports and Total Exports (see Construct Subject)
- Added another illustration of the VOID domain model
- Added URLs for VOID Linksets, e.g. http://vocab.getty.edu/dataset/aat/alignment/lcsh
- Added the full VOID info to the repository (in addition to a designated file), see VOID Deployment. Changed the queries for Dynamic Descriptive Properties to INSERT instead of CONSTRUCT. Added all Prefixes to prefixes.ttl
- Added 2 descriptive info queries at the end of Counting and Descriptive Info
- Produced this document in HTML (inside http://www.getty.edu/research/tools/vocabularies/lod/aat-images.zip), in addition to PDF

1.2.1.5 Version 2.0

15 Jun 2014

Added the second vocabulary: TGN TODO. Surprisingly few changes were needed

- Mapping change: Made Term Characteristics independent of AAT (moved GVP URLs one level up), in preparation for adding TGN.
- Mapping change: Introduced "Extended" versions of GVP Hierarchical Relations and removed gvp:broaderTransitive: ***use gvp:broaderExtended instead of skos:broaderTransitive for appropriate query expansion***. Infer Standard Hierarchical Relations (e.g. iso:broaderGeneric) from these Extended versions (e.g. gvp:broaderGeneric), see BTG, BTP, BTI Inference and ISO Insert Queries
- Mapping change: omitted Undetermined language in GVP Language Tags
- Mapping change: use schema:startDate,endDate for Historic Information and Obsolete Subject, instead of custom properties gvp:startDate,endDate.
- Declared Associative Relationships as sub-properties of skos:related (see Relationship Representation).
- Added more dct:formats to Descriptive Properties (both NTriples and ZIP for total exports, "meta/void" for descriptor, "meta/rdf-schema" for ontology)
- Moved this document to a more logical location (see first page)
- Made the HTML version primary, and made properly named anchors for each section

1.2.1.6 Future Versions

Mapping changes under consideration. Feedback is appreciated:

- Add more TGN External Ontologies TODO
- Map Scope Note to custom class gvp:ScopeNote with rdf:value, instead of xl:Label with xl:literalForm.

1.2.2 External Review Process

See http://www.getty.edu/research/tools/vocabularies/lod/aat_lod_external_advisors.pdf for the list of the individuals who are serving as External Advisors on this project. Most have been recommended by colleagues in our community (e.g., International Terminology Working Group members who are currently translating the AAT into various languages). We ended up inviting a fairly large group because we wanted to make sure that we had expertise in many areas. It has been very important to us that these trusted colleagues had a chance to comment on our ontology choices prior to the release of the datasets.

1.2.3 Providing Feedback

We welcome comments if you find something in this document or our dataset that needs clarification or improvement. We ask specifically for your opinion in several places in this document.

We have established an open community and we welcome collaboration. The preferred way of communicating technical questions and comments is to use the SemanticWeb.com Q&A forum (previously we were getting feedback at an email address but now the forum is preferred). We will be monitoring this forum regularly. The tags to use are: Getty, AAT, TGN, ULAN, CONA. The following links will pre-populate a question form with the appropriate tag:

- http://answers.semanticweb.com/questions/ask/?tags=Getty,AAT
- http://answers.semanticweb.com/questions/ask/?tags=Getty,TGN
- http://answers.semanticweb.com/questions/ask/?tags=Getty,ULAN
- http://answers.semanticweb.com/questions/ask/?tags=Getty,CONA

For questions and comments about editorial content or general information regarding the Getty Vocabularies (not LOD), please contact us at vocab@getty.edu.

1.2.4 Disclaimer

The AAT dataset is provided “as is.” The Getty disclaims all other warranties, either express or implied, including, but not limited to, implied warranties of merchantability and fitness for a particular purpose, with respect to the database. As is true for all Getty vocabularies, the AAT is compiled by the Getty Vocabulary Program from contributions from various contributors, including museums, libraries, archives, bibliographic indexing projects, international translation projects, and others. Not all contributor data complies precisely with the AAT Editorial Guidelines; therefore, absolute consistency in the dataset is not possible. The data is subject to frequent updates and corrections. We anticipate that in the future, the AAT LOD data will be refreshed every two weeks, on the same schedule as the online public data and the data files available to licensees through Web services. For more information about various ways in which the Getty vocabularies may be obtained, see http://www.getty.edu/research/tools/vocabularies/obtain/index.html.

1.3 Abbreviations

Abbrev	Term
AACR2	Anglo-American Cataloging Rules 2
AAT	Art and Architecture Thesaurus
AATNed	AAT Netherlands: the project to make the Dutch translation of the AAT
BIBO	Bibliographic Ontology
DC	Dublin Core (and DC Elements)
DCT	Dublin Core Terms
FOAF	Friend of a Friend ontology
Forest	Ontotext semantic UI framework
FTS	Full Text Search
GRI	Getty Research Institute
GVP	Getty Vocabulary Program
IANA	Internet Assigned Numbers Authority
ISO	International Standardization Organization
ISO 25964	Latest ISO standard on thesauri
LCSH	Library of Congress Subject Headings
LoC	Library of Congress
LOD	Linked Open Data
Lucene	Lucene FTS engine
OWL	Web Ontology Language
OWLIM	Ontotext semantic (RDF) repository
PROV	Provenance (working group, model, ontology)
RDF	Resource Description Framework
RDFa	RDF in Attributes: allows the embedding of RDF data in HTML
RDFS	RDF Schema
SKOS	Simple Knowledge Organization System
SKOS-XL	SKOS Extension for Labels
SPARQL	SPARQL Protocol and RDF Query Language
TGN	Thesaurus of Geographic Names
UI	User Interface
URI	Unified Resource Identifier. Following LOD principles, all our URIs are resolvable, i.e. URLs
URL	Unified Resource Locator

1.4 RDF Turtle

This document uses examples written in Turtle, which is much more readable than other RDF representations such as RDF XML or NTriples.

We use URLs derived from AAT database IDs, so our prefixed names start with a digit.

Examples:

- aat:300198841: a Subject
- aat_term:1000198841-el-Latn: a Term
- aat_source:2000051089-term-1000198841: a Source, namely the "local source" pertaining to this term

The Turtle 1.1 Candidate Recommendation of 19 February 2013 allows the local part of a prefixed name to start with a digit. We provide Export Files in Turtle, RDF XML, NTriples and JSON. To load the Turtle files you need recent parsing tools that support this Turtle 1.1 feature. For example:

- Sesame RIO 2.7.9 (2013-12-18)
- Jena ARQ 2.8.8 (2011-04-21)

Turtle 1.1 names may also include colon ":" (CURIE) and may include other special characters through escaping (e.g. at-sign "\@" and slash "\/"). However we limit the characters used in local names to letters, digits and dash "-" (e.g. the Jena ARQ version cited above does not handle ":" and escaped chars). Because Turtle 1.1 capable tools are not very widely deployed as of the end of 2013, we provide the Total Export files in **NTriples** format.

1.5 Prefixes

The prefixes that we use (both internal and external) are defined below.

The easiest way to find prefixes is the prefix.cc service.

- For example, if you wonder what the rr: prefix means, go to http://prefix.cc/rr. You will quickly find the URL, and whether any other alternatives have been registered.
- If you need the prefix to put in a Turtle file, go directly to http://prefix.cc/rr.ttl. This page allows you to copy it to the clipboard, so you can paste it into a file
- For the GVP Prefixes, we have registered gvp: and aat: in this service (unfortunately it does not support prefixes including "_")

For your convenience, all prefixes that we use are provided at http://www.getty.edu/research/tools/vocabularies/lod/prefixes.ttl.

1.5.1 External Prefixes

We use the following prefixes for External Ontologies. rr: and rrx: (R2RML) are used only during the conversion process. luc: is used for Full Text Search Query

Prefix	URL	Explanation
bibo:	http://purl.org/ontology/bibo/	Bibliography Ontology
dc:	http://purl.org/dc/elements/1.1/	Dublin Core Elements
dct:	http://purl.org/dc/terms/	Dublin Core Terms
foaf:	http://xmlns.com/foaf/0.1/	Friend of a Friend ontology
iso:	http://purl.org/iso25964/skos-thes#	ISO 25946 Thesaurus ontology
luc:	http://www.ontotext.com/owlim/lucene#	Full Text Search predicates using OWLIM's built-in Lucene
owl:	http://www.w3.org/2002/07/owl#	Web Ontology Language
prov:	http://www.w3.org/ns/prov#	Provenance Ontology
rdf:	http://www.w3.org/1999/02/22-rdf-syntax-ns#	Resource Description Framework
rdfs:	http://www.w3.org/2000/01/rdf-schema#	RDF Schema
rr:	http://www.w3.org/ns/r2rml#	Relational to RDF Mapping Language
rrx:	http://purl.org/r2rml-ext/	R2RML extension (rrx:languageColumn)
schema:	http://schema.org/	Schema.org common properties
skos:	http://www.w3.org/2004/02/skos/core#	Simple Knowledge Organization System
skosxl:	http://www.w3.org/2008/05/skos-xl#	SKOS Extension for Labels
xsd:	http://www.w3.org/2001/XMLSchema#	XML Schema Datatypes

1.5.2 Descriptive Prefixes

We use some of the following prefixes for Descriptive Information:

Prefix	URL	Explanation
adms:	http://www.w3.org/ns/adms#	Asset Description Metadata Schema
cc:	http://creativecommons.org/ns#	Creative Commons Rights Expression Language
dc:	http://purl.org/dc/elements/1.1/	Dublin Core Metadata Element Set
dcat:	http://www.w3.org/ns/dcat#	Data Catalog Vocabulary
dct:	http://purl.org/dc/terms/	DCMI Metadata Terms
dctype:	http://purl.org/dc/dcmitype/	DCMI Type Vocabulary
fmt:	http://www.w3.org/ns/formats/	RDF formats used in datasets
sd:	http://www.w3.org/ns/sparql-service-description#	SPARQL Service Description
vaem:	http://www.linkedmodel.org/schema/vaem#	Vocabulary for Attaching Essential Metadata
vann:	http://purl.org/vocab/vann/	Vocabulary for annotating vocabulary descriptions
vcard:	http://www.w3.org/2006/vcard/ns#	vCard (contact info)
vdpp:	http://data.lirmm.fr/ontologies/vdpp#	Vocabulary for Dataset Publication Projects
voaf:	http://purl.org/vocommons/voaf#	Vocabulary of a Friend
voag:	http://voag.linkedmodel.org/voag#	Vocabulary Of Attribution and Governance
void:	http://rdfs.org/ns/void#	Vocabulary of Interlinked Datasets
wdrs:	http://www.w3.org/2007/05/powder-s#	Protocol for Web Description Resources
wv:	http://vocab.org/waiver/terms/	A vocabulary for waivers of rights

1.5.3 GVP URLs and Prefixes

The layout of GVP URLs (both ontology and vocabulary entities) is shown below.

- We use a template notation in the URLs: {m} indicates a numeric identifier, {v} some value, {lang} a language tag
- We have defined prefixes for the most important URLs.
- The AAT subject URIs **aat:{m}** are the only ones that will be used in external datasets, so they are more important than all the other URIs. We keep them as short as possible, allowing shortest Turtle and Sparql.
    - The TGN subject URIs are of the same form **tgn:{m}**. However in your data you will most often want to use the place URIs **tgn:{m}-place**, see TODO.
- URLs shown in roman font represent independent entities. The data of URLs shown in *italic font* is returned together with the owning Subject so that you won't have to chase different URLs to gather the information (these URLs can also be resolved on their own).

Common

Prefix	**URL**	**Explanation**
base	http://vocab.getty.edu	Prepend to all URIs. Returns a home page describing the LOD vocabularies and giving links to the ontology, vocabularies, sample resources, etc
gvp:	/ontology#	GVP Ontology. Uses SKOS, SKOS-XL, ISO 25964, DC, DCT, BIBO, FOAF and PROV. Also holds Associative Relations
gvp_lang:	/language/{lang}	Languages used by the Getty Vocabulary Program. URLs use IANA language tags. Have owl:sameAs links to corresponding concepts in the AAT Languages hierarchy
	/historic/{v}	Historic Flag: Current, Historic or Both
	/term/display/{v}	Term use: for Display or Indexing?
	/term/flag/{v}	Term Flag: Vernacular, Loan Term
	/term/kind/{v}	Term Kind: Neologism, Scientific Name, etc
	/term/POS/{v}	Term Part of Speech: Noun Plural, Noun Singular, etc
	/term/type/{v}	Term Type: Descriptor, Alternate Descriptor, Use For term

AAT

Prefix	**URL**	**Explanation**
aat:	/aat/	AAT vocabulary (skos:ConceptScheme)
	/aat/{m}	AAT subject {m}: Facet, Hierarchy, Guide Term, or Concept
	/aat/{m}-array	Anonymous iso:ThesaurusArray used to represent the ordered concept {m}
	/aat/{m}-list-{n}	rdf:List element for child subject {n} of the skos:OrderedCollection used to represent the ordered subject {m}
aat_contrib:	/aat/contrib/{m}	AAT Contributor {m}
aat_rel:	*/aat/rel/{m}-{type}-{n}*	AAT Relation: from subject {m} to subject {n}, having {type} ("broader" or an associative relation)
aat_rev:	*/aat/rev/{m}*	AAT Revision of a subject
aat_scopeNote:	*/aat/scopeNote/{m}*	AAT Scope Note (definition) {m}
aat_source:	/aat/source/{m}	AAT Source {m}
	/aat/source/{m}-scopeNote-{n}	AAT Source {m} applied to scope note {n}
	/aat/source/{m}-subject-{n}	AAT Source {m} applied to subject {n}
	/aat/source/{m}-term-{n}	AAT Source {m} applied to term {n}
aat_term:	/aat/term/{m}	AAT Term {m}

TGN

Prefix	**URL**	**Explanation**
tgn:	/tgn/	TGN vocabulary (skos:ConceptScheme)
tgn_contrib:	/tgn/contrib/{m}	TGN Contributor {m}
tgn_rel:	*/tgn/rel/{m}-{type}-{n}*	TGN Relation {m}-{type}-{n}: from subject {m} to subject {n}, having {type} ("broader", "placeType", or an associative relation)
tgn_rev:	*/tgn/rev/{m}*	TGN Revision of a subject
tgn_scopeNote:	*/tgn/scopeNote/{m}*	TGN Scope Note (definition) {m}
tgn_source:	/tgn/source/{m}	TGN Source {m}
	/tgn/source/{m}-scopeNote-{n}	TGN Source {m} applied to scope note {n}
	/tgn/source/{m}-subject-{n}	TGN Source {m} applied to subject {n}
	/tgn/source/{m}-term-{n}	TGN Source {m} applied to term {n}
tgn_term:	/tgn/term/{m}	TGN Term {m}

Please note that URLs including the parasitic word "/resource" (e.g. http://vocab.getty.edu/resource/aat/{m}) are NOT valid GVP URLs and you should not use them in your applications, nor make statements about them. This word was used in earlier versions of the http://vocab.getty.edu website due to the internal architecture of Forest UI. People were copying them from the browser address bar and sharing them in discussions, so we went to the trouble of reworking Forest to remove this word, in order to avoid confusion.

1.6 Semantic Resolution

All GVP and AAT URLs resolve, returning human or machine readable content through content negotiation.

- We followed the recommendation Cool URIs for the Semantic Web
- We followed Best Practice Recipes for Publishing RDF Vocabularies
- We validated the resolution with Vapour

Example about an AAT subject: aventurine (quartz)

- http://vocab.getty.edu/aat/300011154 : semantic URI, content-negotiated using 303 redirect
- http://vocab.getty.edu/aat/300011154.html : Forest HTML page (application/xhtml+xml).
- http://vocab.getty.edu/aat/300011154.rdf : application/rdf+xml
- http://vocab.getty.edu/aat/300011154.ttl : text/turtle
- http://vocab.getty.edu/aat/300011154.nt : NTriples
- http://vocab.getty.edu/aat/300011154.json : JSON

Example about a contributor: AATNed

- http://vocab.getty.edu/aat/contrib/10000205: semantic URI, content-negotiated using 303 redirect
- http://vocab.getty.edu/aat/contrib/10000205.html: Forest HTML page (application/xhtml+xml)
- http://vocab.getty.edu/aat/contrib/10000205.rdf : application/rdf+xml
- http://vocab.getty.edu/aat/contrib/10000205.ttl : text/turtle
- http://vocab.getty.edu/aat/contrib/10000205.nt : NTriples
- http://vocab.getty.edu/aat/contrib/10000205.json : JSON

Example about a source: Grove art online

- http://vocab.getty.edu/aat/source/2000049829: semantic URI, content-negotiated using 303 redirect
- http://vocab.getty.edu/aat/source/2000049829.html: Forest HTML page (application/xhtml+xml)
- http://vocab.getty.edu/aat/source/2000049829.rdf : application/rdf+xml
- http://vocab.getty.edu/aat/source/2000049829.ttl : text/turtle

- http://vocab.getty.edu/aat/source/2000049829.nt : NTriples
- http://vocab.getty.edu/aat/source/2000049829.json : JSON

We use suffixes (file extensions) instead of prefixes (folders), since such URIs:

- Emphasize they all are about one semantic entity, instead of being put in "different folders", see Hierarchical URIs Pattern
- Are more hackable (easier to add suffix than spot the prefix), see Patterned URIs Pattern
- Provide correct file extensions for downloading

1.7 External Ontologies

Our mapping uses a number of external ontologies (as listed in External Prefixes):

- SKOS, SKOSXL, ISO 25964 for representing thesaurus info;
- DC, DCT for common properties;
- BIBO, FOAF for sources and contributors;
- PROV for revision history;
- RDF, RDFS, OWL, XSD for system properties;
- R2RML for implementing the conversion.

The current versions of external ontologies should be loaded in the semantic repository. We use RDF instead of Turtle versions, to avoid the addition of spurious prefixes to he repository (e.g. skos.ttl defines an empty prefix ":"):

- SKOS: http://www.w3.org/2004/02/skos/core.rdf
- SKOS-XL: http://www.w3.org/2008/05/skos-xl.rdf
- ISO 25964: iso-thes.rdf obtained with

```
wget --header accept:application/rdf+xml http://purl.org/iso25964/skos-thes#
```

We don't load the other ontologies since they don't make useful inferences for us. In particular, DCT makes a lot of highly irrelevant inferences, e.g. dct:source infers dct:relation and dc:relation.

Brief notes about some of these ontologies follow.

1.7.1 DC and DCT

Dublin Core is an often-used ontology for defining basic metadata (e.g. title, creator, created, modified, issued, source, language, etc).

It defines two metadata sets: DC Elements (older, often denoted simply DC and using prefix dc:) and DC Terms (newer, often denoted DC and using prefix dct:). Their distinguishing characteristics are:

- DC properties allow any values, literal or URI alike. DCT properties are more strict, and a lot of them require URI.
- DC properties stand alone; DCT properties are often defined as sub-properties of DC (or other DCT properties)

By way of example, the two corresponding properties dc:language and dct:language are defined as if:

```
dc:language a rdf:Property .
dct:language a owl:ObjectProperty; rdfs:subPropertyOf dc:language; rdfs:range dct:LinguisticSystem .
```

- dc:language can take any value, either literal or URI
- dct:language takes only URIs, infers dc:language, and infers that the URI has class dct:LinguisticSystem.

We use DC/DCT for various common properties, e.g. dc:identifier, dct:source, dct:contributor, dct:created, dct:modified. If both a DC and DCT property fit a purpose, we use the DCT property if the target is a URL.

1.7.2 SKOS and SKOS-XL

SKOS is a widely used ontology for representing thesauri. SKOS-XL allows you to represent labels as nodes, and attach additional information

We assume the reader has basic knowledge of SKOS. You can consult the following sources:

- SKOS Primer
- SKOS Reference

- SKOS-XL in Primer
- SKOS-XL in Reference

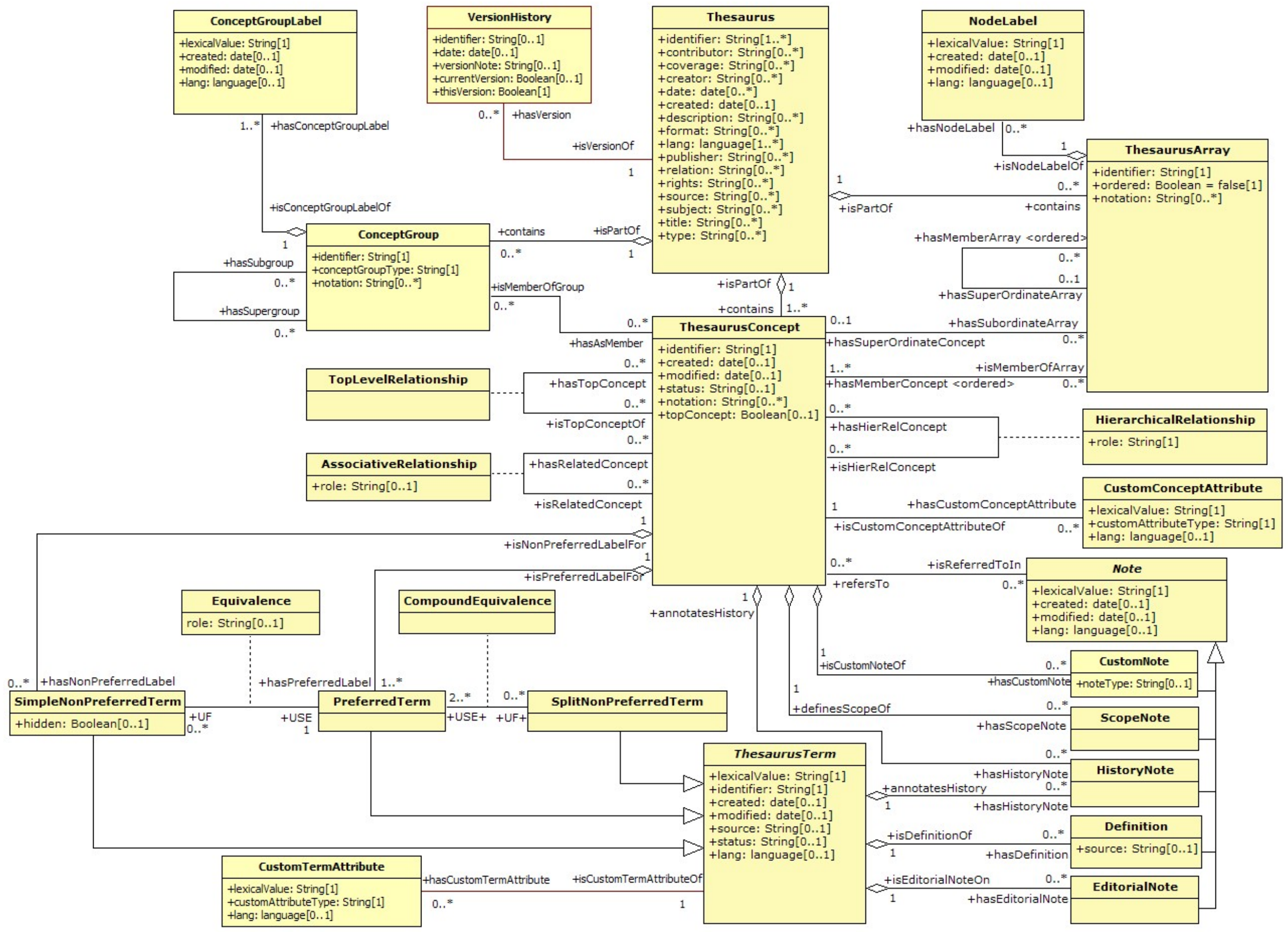
The SKOS standard, as any other standard, is the result of certain tradeoffs and timeline constraints. Therefore various issues and topics that were proposed for consideration did not make it into the final standard. The semantic representation of the GVP thesauri pushes the SKOS envelope in many cases, so it is useful to learn about approaches that go beyond the current SKOS standard.

We found this paper very useful: Key choices in the design of Simple Knowledge Organization System (SKOS), Journal of Web Semantics, May 2013.

See also sections SKOS Inference and SKOS-XL Inference in this document.

1.7.3 ISO 25964

ISO 25964 - the international standard for thesauri and interoperability with other vocabularies is the latest ISO standard on thesauri. The ISO 25964 domain model is shown below:



A lot of it corresponds to SKOS/SKOS-XL, see Correspondence between ISO 25964 and SKOS/SKOS-XL Models. But it has additional constructs not covered by SKOS. In particular, skos:Collection has these limitations:

- you can't put them under a Concept
- you can't say explicitly which are Top Collections in a scheme
- you don't have inverse/transitive versions of skos:member

We use iso:ThesaurusArray, which is a subclass of skos:Collection but can be put under a Concept using iso:superOrdinate (and/or its inverse iso:subordinateArray), see Sorting with Thesaurus Array.

Despite the risks inherent in early adoption of new technology, Getty willingly undertakes this trail-blazing role, because the ISO ontology allows a faithful representation of GVP data, and in order to promote the adoption and deployment of the standard, which is the way to make technical progress.

We provided implementation experience to the ISO technical committee, and contributed suggestions and fixes to the iso-thes ontology (first published on 30 Sep 2013 at the public-esw-thes@w3.org mailing list). To the best of our knowledge, this application to AAT is the first industrial use of ISO 25964.

1.7.4 BIBO and FOAF

The Bibliographic Ontology (BIBO) is used by various library projects, including the British National Bibliography. It is described at http://bibliontology.com, and is kept at GitHub. The definition is available in XML OWL, N3, OWLDoc. We use BIBO to represent Source information: a source is represented as bibo:Document, and additionally as bibo:DocumentPart if there is location information.

The Friend of a Friend ontology (FOAF) is a well-known ontology for people, organizations, contacts, etc. We use FOAF to represent Contributor information: a contributor is represented as foaf:Agent.

1.7.5 PROV

Provenance is information about entities, activities, and people involved in producing a piece of data or thing, which can be used to form assessments about its quality, reliability or trustworthiness.

PROV is a formalization of Provenance by the Provenance Working Group at W3C (PROVG). It defines a model, corresponding serializations, definitions supporting mapping to other provenance models, and examples how to use PROV. The PROV family of documents includes the following:

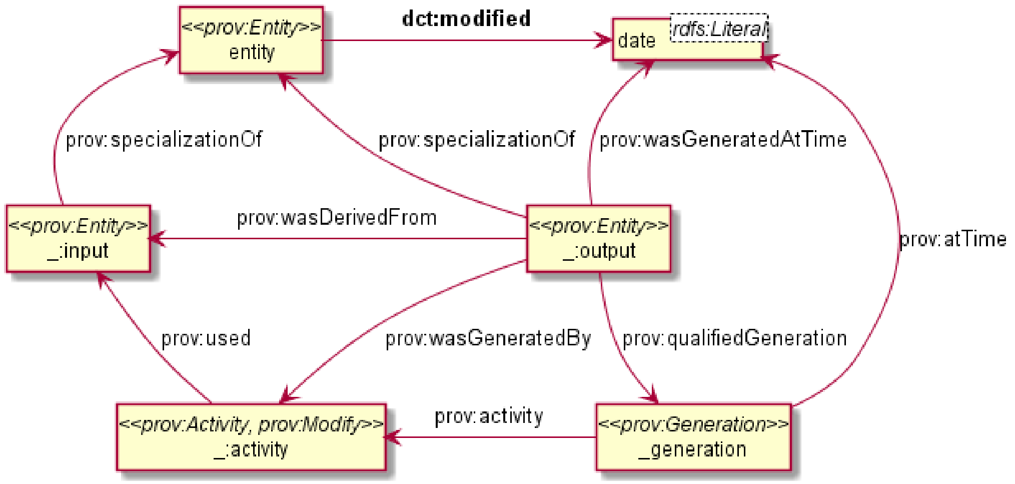
- PROV-OVERVIEW, an overview of the PROV family of documents
- PROV-PRIMER, a primer for the PROV data model
- PROV-DM, the PROV data model
- PROV-O, the PROV ontology, an OWL2 ontology allowing the mapping of the PROV data model to RDF.
- PROV-XML, an XML schema for the PROV data model
- PROV-N, a notation for provenance aimed at human consumption
- PROV-CONSTRAINTS, a set of constraints applying to the PROV data model
- PROV-AQ, mechanisms for accessing and querying provenance
- PROV-DICTIONARY introduces a specific type of collection, consisting of key-entity pairs
- PROV-DC provides a mapping between Dublin Core Terms and PROV-O
- PROV-SEM, a declarative specification in terms of first-order logic of the PROV data model
- PROV-LINKS: introduces a mechanism to link across bundles of provenance info

The PROV-O and PROV-DC ontologies are available as: prov-o.ttl, prov-dc-refinements.ttl.

The PROV-DC mapping illustrates the complexity of PROV best. Below are two examples:

1.7.5.1 dct:modified

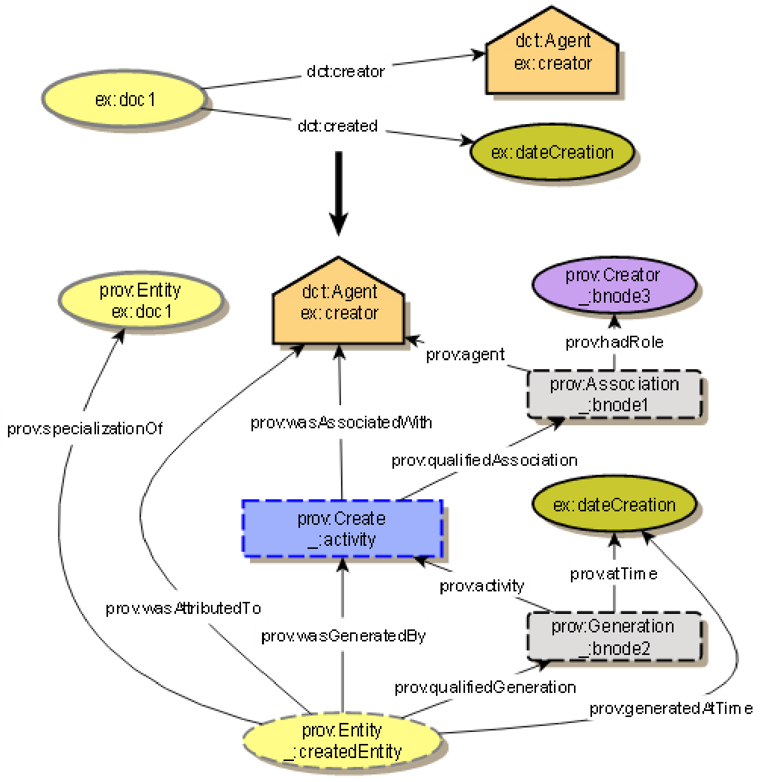
A single statement dct:modified is mapped to a network of 6 nodes and 9 statements:



PROV considers that the prov:Modify activity uses an unknown old entity (**_:input**) and generates an unknown new entity (**_:output**), both being specializations of the **entity** under consideration. Furthermore, we need to use a prov:Generation node to be able to use prov:atTime and reflect accurately that the modification is in fact a prov:InstantaneousEvent.

1.7.5.2 dct:creator+dct:created

Two statements dct:creator and dct:created are mapped to a network of 8 nodes and 11 statements:



1.8 GVP Ontology

The GVP ontology includes various classes, properties and individuals (values) used in the mapping. The prefix **gvp:** stands for "Getty Vocabulary Program". We considered using the more telling prefix **getty:** but decided against it because other Getty institutions (e.g. the Getty Museum) may start publishing LOD soon. Most of the classes and properties are applicable not only to AAT, but also to the other GVP vocabularies to be published in the future.

The ontology is documented at the "namespace document" http://vocab.getty.edu/ontology and is summarized below. You can also get the ontology in XML/RDF and Turtle, either using content negotiation (see Semantic Resolution), or using the direct links http://vocab.getty.edu/ontology.rdf and http://vocab.getty.edu/ontology.ttl respectively. You can also Explore the Ontology with SPARQL queries.

The context where these classes and properties are used is best seen at Semantic Overview.

- Subject Types: gvp:Facet, gvp:Hierarchy, gvp:GuideTerm, gvp:Concept, gvp:ObsoleteSubject.
  These are implemented as subclasses of skos:Concept, skos:Collection, iso:ThesaurusArray.
- Subject Hierarchy relations: gvp:broader, gvp:narrower, gvp:broaderExtended, gvp:narrowerExtended, gvp:broaderPreferred, gvp:broaderPreferredExtended, gvp:broaderNonPreferred, gvp:broaderGeneric, gvp:broaderPartitive, gvp:broaderInstantial, gvp:broaderGenericExtended, gvp:broaderPartitiveExtended, gvp:broaderInstantialExtended.
  These are analogous to skos:broader and friends, but apply to any subject type, and introduce finer distinctions.
- Properties gvp:prefLabelGVP, gvp:prefLabelLoC: most often these are "parallel to" (used with) skosxl:prefLabel
- Other Subject properties: gvp:parentString, gvp:parentStringAbbrev,
- Term Characteristics: gvp:termKind, gvp:termDisplay, gvp:termType, gvp:termPOS, gvp:termFlag
- Other Term relations: gvp:contributorPreferred, gvp:contributorNonPreferred, gvp:contributorAlternatePreferred (sub-properties of dct:contributor); gvp:sourcePreferred, gvp:sourceNonPreferred, gvp:sourceAlternatePreferred (sub-properties of dct:source)
- Sort Order (applies to Subject and Term): gvp:displayOrder
- Historic Information properties (apply to Subject and Term): gvp:historicFlag, schema:startDate, schema:endDate
- Finally, Associative Relations are defined as properties in the GVP ontology (there is a large number of these)

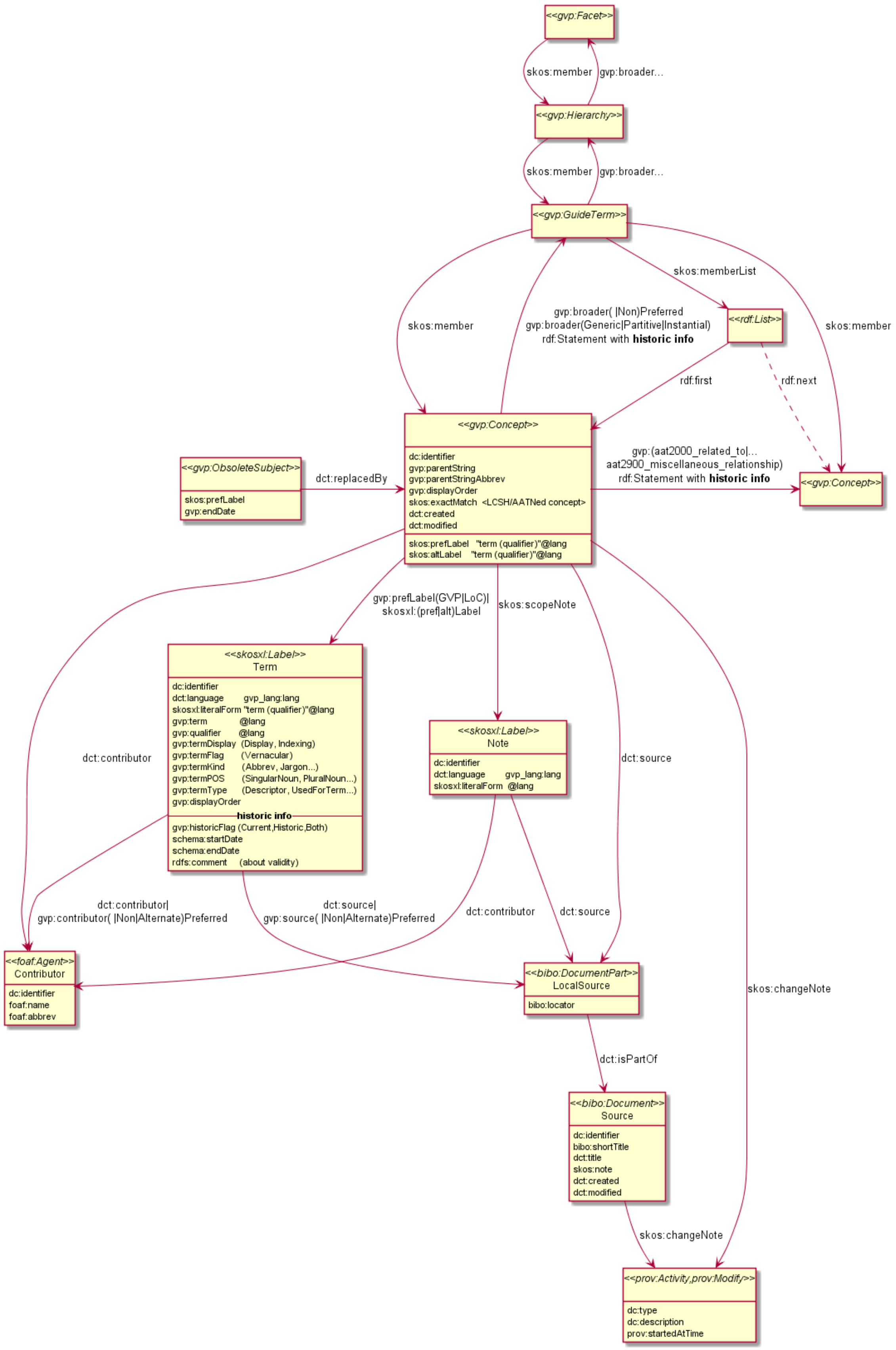
Acknowledgements:

- We created the ontology documentation (namespace document) with Parrot. Note that going to the URL of a class/property (e.g. http://vocab.getty.edu/ontology#aat2000_related_to) will jump directly to the definition of that class/property. The documentation also includes embedded RDFa "about" attributes.
- We intend to validate the ontology using OOPS! (the OntOlogy Pitfall Scanner!).

2 Semantic Representation

2.1 Semantic Overview

The diagram below provides an overview of the semantic representation


<<gvp:Facet>>
skos:member
gvp:broader...
<<gvp:Hierarchy>>
skos:member
gvp:broader...
<<gvp:GuideTerm>>
skos:memberList
skos:member
gvp:broader(|Non)Preferred
gvp:broader(Generic|Partitive|Instantial)
rdf:Statement with **historic info**
<<rdf:List>>
skos:member
rdf:first
rdf:next
<<gvp:Concept>>
dc:identifier
gvp:parentString
gvp:parentStringAbbrev
gvp:displayOrder
skos:exactMatch <LCSH/AATNed concept>
dct:created
dct:modified
skos:prefLabel "term (qualifier)"@lang
skos:altLabel "term (qualifier)"@lang
<<gvp:ObsoleteSubject>>
skos:prefLabel
gvp:endDate
dct:replacedBy
gvp:(aat2000_related_to|...
aat2900_miscellaneous_relationship)
rdf:Statement with **historic info**
<<gvp:Concept>>
gvp:prefLabel(GVP|LoC)|
skosxl:(pref|alt)Label
skos:scopeNote
<<skosxl:Label>>
Term
dc:identifier
dct:language gvp_lang:lang
skosxl:literalForm "term (qualifier)"@lang
gvp:term @lang
gvp:qualifier @lang
gvp:termDisplay (Display, Indexing)
gvp:termFlag (Vernacular)
gvp:termKind (Abbrev, Jargon...)
gvp:termPOS (SingularNoun, PluralNoun...)
gvp:termType (Descriptor, UsedForTerm...)
gvp:displayOrder
historic info
gvp:historicFlag (Current,Historic,Both)
schema:startDate
schema:endDate
rdfs:comment (about validity)
<<skosxl:Label>>
Note
dc:identifier
dct:language gvp_lang:lang
skosxl:literalForm @lang
dct:contributor
dct:source
dct:contributor|
gvp:contributor(|Non|Alternate)Preferred
dct:source|
gvp:source(|Non|Alternate)Preferred
dct:contributor
dct:source
<<foaf:Agent>>
Contributor
dc:identifier
foaf:name
foaf:abbrev
<<bibo:DocumentPart>>
LocalSource
bibo:locator
skos:changeNote
dct:isPartOf
<<bibo:Document>>
Source
dc:identifier
bibo:shortTitle
dct:title
skos:note
dct:created
dct:modified
skos:changeNote
<<prov:Activity,prov:Modify>>
dc:type
dc:description
prov:startedAtTime

The diagram shows pretty much everything but glosses over some details:

- Subjects are the main entities of GVP vocabularies. They have the following Subject Types: Facet, Hierarchy, Guide Term, Concept (arranged in a hierarchy), and Obsolete Subject (details are glossed over in the diagram).
    - They are implemented using skos:Collection and iso:ThesaurusArray (for the first 3) and skos:Concept (for gvp:Concept)
    - skos:OrderedCollection and rdf:List is also used when a Subject's children are ordered (see Sorting with Thesaurus Array)
- Subjects (except Obsolete) have the following info (exactMatch and Associative Relations apply to Concepts only):
    - Connected to the aat: ConceptScheme using skos:inScheme (not shown on the diagram). Concepts also have skos:topPropertyOf as appropriate
    - dc:identifier (see Identifiers)
    - gvp:parentString and gvp:parentStringAbbrev
    - gvp:displayOrder (see Sort Order)
    - Historic Information
    - skos:exactMatch to other thesauri (see Alignment)
    - gvp:prefLabelGVP, gvp:prefLabelLoC, skosxl:prefLabel or skosxl:altLabel links to Terms; and skos:prefLabel, skos:altLabel as dumbed-down versions of these links (SKOS-XL Inference)
    - skos:scopeNote links to Scope Notes
    - dct:source links to Source (can be to a LocalSource as shown, or directly to a global Source)
    - dct:contributor links to Contributor
- Subject Hierarchy relations come in several varieties: Preferred|NonPreferred and Generic|Partitive|Instantial (details are glossed over in the diagram).
    - They can be accessed uniformly through gvp:broader, gvp:narrower and their transitive variants
    - They are implemented using skos:narrower, skos:member, skos:memberList, iso:subordinateArray (going down) and skos:broader, iso:superOrdinate (going up). The transitive variants skos:narrowerTransitive, skos:broaderTransitive are also provided
- Associative Relations (properties numbered from gvp:aat2000_* to gvp:aat2900_*) provide lateral relations between subjects.
    - Both Hierarchical and Associative relations may carry Historic Information attached to a rdf:Statement
- Obsolete Subjects provide some continuity to clients who have used such subjects in their data. They have little info: only skos:prefLabel, schema:endDate (when it was discontinued), and dct:replacedBy (if it was merged to another subject).
- Terms provide multilingual subject labels and carry the following information:
    - dc:identifier (see Identifiers)
    - Literals: gvp:term, gvp:qualifier and skosxl:literalForm (being "term (qualifier)", or equal to term if there's no qualifier). All these have the same language tag (@lang)
    - dct:language (Language). Coincidentally, if the language is gvp_lang:<lang>, then the URL of the Term is aat_term:<identifier>-<lang>.
    - Enumerated Term Characteristics: termDisplay, termFlag, termKind, termPOS and termType
    - gvp:displayOrder (see Sort Order)
    - Historic Information
    - Links to Source (can be to a LocalSource as shown, or directly to a global Source). These links can be plain dct:source; or sub-properties thereof, specifying whether the term is Preferred, NonPreferred or AlternatePreferred for the source
    - Links to Contributor. Similar to Source links, these can be plain dct:contributor, or sub-properties thereof
- Scope Notes provide multilingual subject definitions and carry the following information:

  - dc:identifier (see Identifiers)
  - Literal: skosxl:literalForm with a language tag (@lang)
  - dct:language (Language).
  - dct:source links to Source (can be to a LocalSource as shown, or directly to a global Source)
  - dct:contributor links to Contributor
- Contributors are foaf:Agents and have this info: dc:identifier, foaf:name and foaf:abbrev
- Sources (global sources) are bibo:Documents and have this info: dc:identifier, bibo:shortTitle, dc:title and skos:note
  - When an entity (Subject, Term or Scope Note) is cited in a particular spot in a source, then we have a Local Source with bibo:locator giving the spot, and dct:isPartOf pointing to the global source

2.2 Subject

Subjects are the main entities (units of thought) in the GVP vocabularies. Subjects (except Obsolete) have the following info, described in the respective sections:

- Subjects are connected to the aat: ConceptScheme using skos:inScheme.
- Subject Hierarchy: threads the subjects using custom (gvp:) properties. Standard skos: and iso: properties are also provided
- Associative Relations: apply to Concepts only
- dc:identifier (see Identifiers)
- gvp:displayOrder (see Sort Order)
- Historic Information about historic applicability
- skos:exactMatch to other thesauri (see Alignment): applies to Concepts only
- skos:scopeNote links to Scope Notes
- dct:source links to Source (can be to a LocalSource as shown, or directly to a global Source)
- dct:contributor links to Contributor

Subjects also have these specific properties:

Property	**range**	**Definition**
gvp:prefLabelGVP	skosxl:Label	Term preferred by the Getty Vocabulary Program. The language is usually English. Applicable to AAT, ULAN, TGN. Most often used with skosxl:prefLabel
gvp:prefLabelLoC	skosxl:Label	Term preferred by Library of Congress, thus used for cataloging according to AACR2. Applicable to AAT and ULAN. Most often used with skosxl:prefLabel
skosxl:prefLabel	skosxl:Label	Preferred term (descriptor)
skosxl:altLabel	skosxl:Label	Non-preferred term (AlternateDescriptor or UsedForTerm)
gvp:parentString	literal	Preferred labels of the subject's preferred ancestors, listed bottom up. Useful to show the subject's full context. E.g. for 300226882 "baking dishes" is: "bakeware, <vessels for cooking food>, <containers for cooking food>, <culinary containers>, <containers by function or context>, containers (receptacles), Containers (Hierarchy Name), Furnishings and Equipment (Hierarchy Name), Objects Facet"
gvp:parentStringAbbrev	literal	Same but skips middle levels for brevity. E.g. for 300226882 "baking dishes" is: "bakeware, <vessels for cooking food>, ... Furnishings and Equipment (Hierarchy Name)"

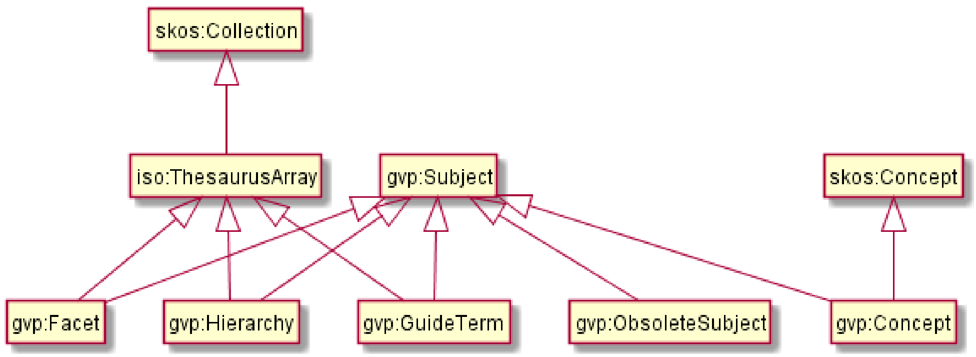
Subjects also have data properties skos:prefLabel, skos:altLabel as dumbed-down versions of skosxl:prefLabel and skosxl:altLabel (SKOS-XL Inference)

2.2.1 Subject Types

GVP uses different types of subject to construct the levels of the Subject Hierarchy:

Type	Definition	Example
Facet	One of the major divisions of the AAT.	Objects Facet
Hierarchy Name	The top of a hierarchy. Not used for indexing or cataloguing.	Containers (Hierarchy Name)
Guide Term	Place holder to create a level in the hierarchy. Not used for indexing or cataloguing.	<vessels for serving and consuming food>
Concept	Proper concept. Used for indexing and cataloguing.	"rhyta"
Obsolete Subject	Obsolete: moved out of the publishable hierarchy, or merged to another	300375205 "shranks" was merged to 300039264 "schranks" (reflected by dct:isReplacedBy)

We have introduced GVP classes for each type, and arranged them in the following hierarchy:



- All are subclasses of **gvp:Subject**, to allow the user to find all AAT subjects with a single query
- gvp:Facet, gvp:Hierarchy and gvp:GuideTerm are implemented as a subclass of **iso:ThesaurusArray** (which itself is a subclass of skos:Collection) since they can hold other subjects, but cannot be used for indexing
- gvp:Concept is implemented as a subclass of **skos:Concept**, since they are used for indexing
- gvp:ObsoleteConcept is not a subclass of any standard class, since normal thesaurus operations should NOT use nor display them

When a Subject's children are ordered, it is also declared skos:OrderedCollection (see Sorting with Thesaurus Array)

2.3 Subject Hierarchy

The GVP hierarchy includes subjects other than Concepts (see Subject Types); distinguishes between a Preferred parent and optional NonPreferred parents, and makes distinctions between broaderGeneric, broaderPartitive and broaderInstantial.

There is a bunch of hierarchical relation properties, which we explain in detail below, including which properties are derived from which others. Then we explain how the properties are used across the Hierarchy Structure, and which property is used for which subject types.

2.3.1 Standard Hierarchical Relations

SKOS and ISO 25964 provide a number of hierarchical relations. We use the following (**d** shows the direction up/down):

Relation	**d**	**Domain**	**Range**	**Description**
skos:broader	▲	skos:Concept	skos:Concept	Parent concept of a concept
iso:broaderGeneric	▲	skos:Concept	skos:Concept	Parent in the case of Genus/Species relation
iso:broaderPartitive	▲	skos:Concept	skos:Concept	Parent in the case of Part/Whole relation
iso:broaderInstantial	▲	skos:Concept	skos:Concept	Parent in the case of Kind/Instance relation
skos:broaderTransitive	▲	skos:Concept	skos:Concept	Ancestor concepts (transitive version of broader)
iso:superOrdinate	▲	iso:ThesaurusArray	skos:Concept	Parent concept of array
skos:narrower	▼	skos:Concept	skos:Concept	Children concepts of a concept
skos:narrowerTransitive	▼	skos:Concept	skos:Concept	Descendant concepts (transitive version of narrower)
skos:member	▼	iso:ThesaurusArray	skos:Concept	Children concepts/arrays of array. See skos:member Structure for an illustration. skos:memberList is also used if the array is ordered, see skos:memberList Structure
iso:subordinateArray	▼	skos:Concept	iso:ThesaurusArray	Children arrays of a concept

2.3.2 GVP Hierarchical Relations

The standard relations have certain limitations:

- Different properties are used for different nodes in the hierarchy (Subject Types), which does not allow you to access the hierarchy uniformly
- iso:broaderGeneric, iso:broaderPartitive, iso:broaderInstantial (commonly known as BTG, BTP, BTI) are important distinctions of the broader relation. These apply to skos:Concept only, but GVP needs to use them for other Subject Types as well.
- These ISO properties are declared sub-properties of skos:broader, which itself is a sub-property of skos:broaderTransitive, so all these relations unconditionally contribute to skos:broaderTransitive. However, composing these relations is not appropriate in some cases, e.g. *Sofia* BTP *Bulgaria* BTI *country*, but *Sofia* BTI *country* is false: *Sofia* BTI *city* (or *inhabited place*), and there is no broader relation between *city* and *country*. So skos:broaderTransitive is not meaningful in these cases.

To overcome these limitations, we introduce a number of custom (GVP) relations, all of which are defined across all subject types (i.e. domain and range is gvp:Subject). The paper Compositionality of ISO 25964 Hierarchical Relations (V.Alexiev, J.Lindenthal, A.Isaac, May 2014, submitted) discusses the last point (meaningful closure) in detail:

- Introduces "Extended" relations (gvp:broaderGenericExtended, gvp:broaderPartitiveExtended, gvp:broaderInstantialExtended, denoted for brevity BTGE, BTPE, BTIE), meant to be meaningful closures of BTG, BTP, BTI.
- Proposes specific compositionality rules to derive the Extended relations as specific chains, see BTG, BTP, BTI Inference.
- Defines gvp:broaderExtended (and its inverse gvp:narrowerExtended) as a disjunction of BTGE, BTPE, BTIE. ***Thesaurus connoisseurs should use gvp:broaderExtended instead of skos:broaderTransitive for proper query expansion.***
- Derives the ISO BTG, BTP, BTI relations from BTGE, BTPE, BTIE connecting skos:Concepts directly, see ISO Insert Queries.

(**d** shows the direction up/down)

Relation	d	Name	Description
gvp:broader	▲	Parents	Each broader is also Preferred | NonPreferred and Partitive | Instantial | Generic
gvp:narrower	▼	Children	Inverse of gvp:broader
gvp:broaderPreferred	▲	Preferred Parent	Main parent, e.g. *baking dishes* BTG *bakeware* is preferred. May have two, e.g. *Sofia* BTP *Bulgaria* and *Sofia* BTI *inhabited place* are **both** preferred.
gvp:broaderNonPreferred	▲	Non-Preferred Parents	Auxiliary parents, e.g. *baking dishes BTG dishes (vessels)* and *Sofia* BTI *cultural capital*[1] are non-preferred. Very often there are several non-preferred parents.
gvp:broaderGeneric	▲	Parent (Generic)	BTG (Genus/Species, "is a") relation, e.g. *calcite* (AAT) BTG *mineral* (AAT)
gvp:broaderPartitive	▲	Parent (Partitive)	BTP (Part/Whole, "part of") relation, e.g. *Tuscany* (TGN) BTP *Italy* (TGN)
gvp:broaderInstantial	▲	Parent (Instantial)	BTI (Kind/Instance, "example of") relation, eg *Rembrandt van Rijn* (ULAN) BTP *Painter* (AAT)
gvp:broaderGenericExtended	▲	Ancestors (Generic)	Meaningful closure of gvp:broaderGeneric
gvp:broaderPartitiveExtended	▲	Ancestors (Partitive)	Meaningful closure of gvp:broaderPartitive
gvp:broaderInstantialExtended	▲	Ancestors (Instantial)	Meaningful closure of gvp:broaderInstantial
gvp:broaderExtended	▲	Appropriate ancestors	Meaningful closure of gvp:broader for query expansion. Use this, not skos:broaderTransitive
gvp:narrowerExtended	▼	Appropriate descendants	Meaningful closure of gvp:narrower for query expansion. Use this, not skos:narrowerTransitive
gvp:broaderPreferredExtended	▲	Preferred Ancestors	Meaningful closure of gvp:broaderPreferred

2.3.3 Hierarchy Structure

The structure of the subject hierarchy can be symbolized as **F>H>G&C**, i.e.

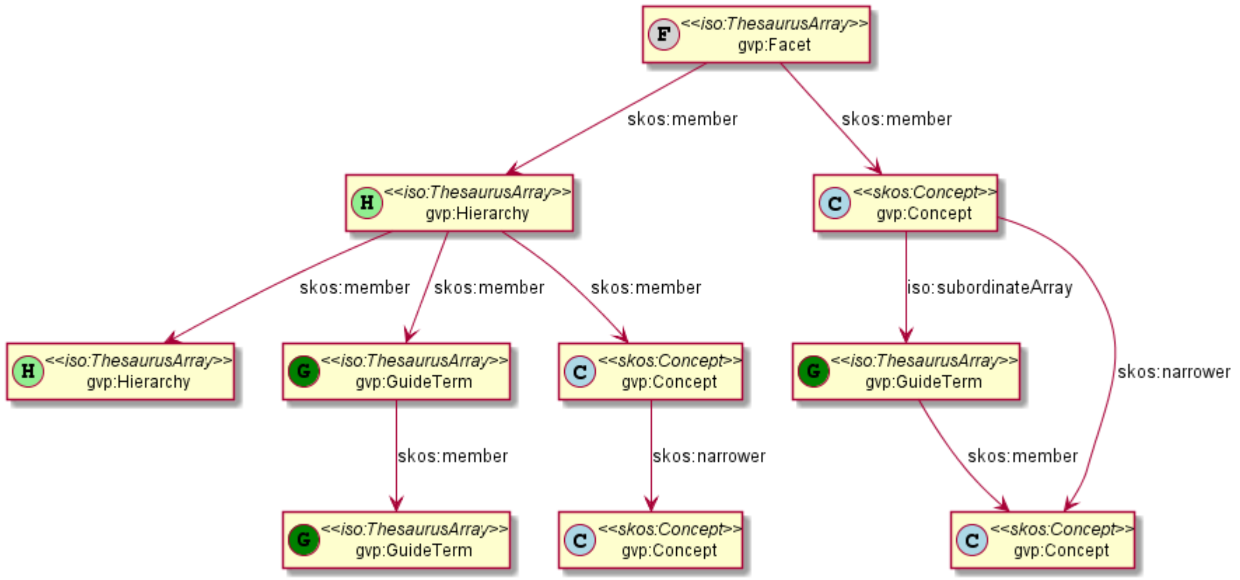
- Facets are above Hierarchies
- Hierarchies are above Guide Terms and Concepts
- Guide Terms and Concepts can be intermixed. There are many examples of G under C. (Note: during vocabulary evolution, G sometimes transitions to C)

The following table shows for each Subj1, what Subj2 can be nested under it (Facet cannot be nested under anything). In each cell we give an example of Subj2, and the linking property (e.g. Concept→ iso:subordinateArray→ GuideTerm).

[1] Hopefully in 2019, see http://www.sofia2019.bg/en

Subj1\Subj2	Hierarchy	Guide Term	Concept
Facet	Living Organisms skos:member		agents (general) skos:member
Hierarchy	Visual Works (Hierarchy) skos:member	<organisms by activity> skos:member	visual works skos:member
Guide Term		<Early Western World> skos:member	Mediterranean (Early Western World) skos:member
Concept		<ancient Italian styles and periods> iso:subordinateArray	Old Hittite Kingdom skos:narrower

The following diagram illustrates the same nesting possibilities:

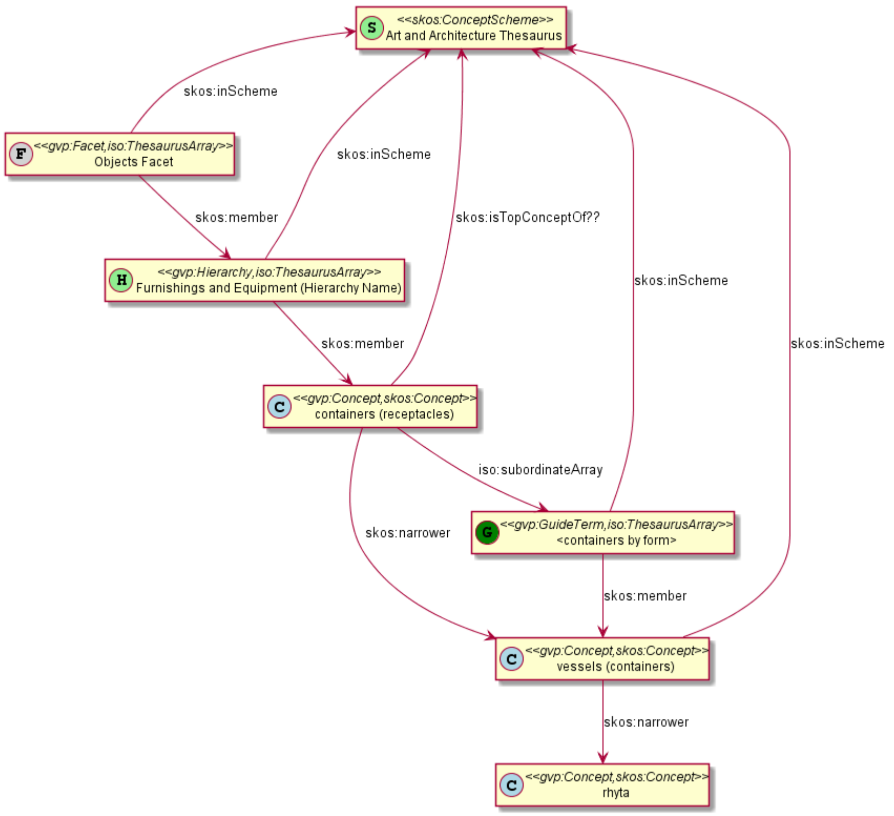

F <<iso:ThesaurusArray>> gvp:Facet
skos:member
skos:member
H <<iso:ThesaurusArray>> gvp:Hierarchy
C <<skos:Concept>> gvp:Concept
skos:member
skos:member
skos:member
iso:subordinateArray
skos:narrower
H <<iso:ThesaurusArray>> gvp:Hierarchy
G <<iso:ThesaurusArray>> gvp:GuideTerm
C <<skos:Concept>> gvp:Concept
G <<iso:ThesaurusArray>> gvp:GuideTerm
skos:member
skos:narrower
skos:member
G <<iso:ThesaurusArray>> gvp:GuideTerm
C <<skos:Concept>> gvp:Concept
C <<skos:Concept>> gvp:Concept

You can see that the standard representation is not uniform (uses different properties in the different cases). You may also notice the right-most link skos:narrower: although it's not in the original hierarchy, we insert it, so as to thread the Concepts through the hierarchy. We call this a "thread-through" skos:narrower.

Keep in mind that adjacent nodes are also connected uniformly by the custom properties listed in the previous section (e.g. gvp:narrower going down). Because skos:narrower connects only Concepts but "jumps levels" as in the example above, it is neither a sub-property nor a super-property of gvp:narrower.

2.3.4 Top Concepts

Consider this example from the AAT hierarchy. Legend: S=scheme, F=facet, H=hierarhchy, G=guide term, C=concept. (Some levels are skipped for brevity, inverses and inferred properties are not shown). Every subject in the hierarchy has a link to the AAT ConceptScheme: skos:inScheme aat:


S <<skos:ConceptScheme>> Art and Architecture Thesaurus
skos:inScheme
F <<gvp:Facet,iso:ThesaurusArray>> Objects Facet
skos:inScheme
skos:member
skos:isTopConceptOf??
skos:inScheme
H <<gvp:Hierarchy,iso:ThesaurusArray>> Furnishings and Equipment (Hierarchy Name)
skos:inScheme
skos:member
C <<gvp:Concept,skos:Concept>> containers (receptacles)
iso:subordinateArray
skos:narrower
G <<gvp:GuideTerm,iso:ThesaurusArray>> <containers by form>
skos:member
C <<gvp:Concept,skos:Concept>> vessels (containers)
skos:narrower
C <<gvp:Concept,skos:Concept>> rhyta

Consider the Concept "containers (receptacles)": it has a thread-through skos:narrower to "vessels (containers)". Furthermore, it is a Top Concept, because there are only Facets, Hierarchies and Guide Terms above it.

Many SKOS thesauri use skos:topConceptOf (a sub-property of skos:inScheme) to link such Concepts to the ConceptScheme. Even flat concept lists do that, e.g. LoC's MARC Relators are declared topConceptOf http://id.loc.gov/vocabulary/relators (see HTML and RDF).

In a previous version of AAT we did the same, but it has some counter-intuitive consequences. E.g. aat:300054031 "drawing (metalworking)" is a top concept, although it's nested 8 levels deep:

- <metal forming processes and techniques>, <metalworking processes and techniques>, <metalworking and metalworking processes and techniques>, <processes and techniques by material>, <processes and techniques by specific type>, <processes and techniques>, Processes and Techniques, Activities Facet

The reason is that none of its ancestors is a concept: going up, there are 6 Guide Terms, then a Hierarchy, and finally a Facet.

skos:hasTopConcept (the inverse of skos:topConceptOf) is meant as a navigation-enabler, to provide "entry points" in the hierarchy (see SKOS Primer 2.5 Concept Schemes). In AAT the Facets are such entry points, and it is dubious that top concepts would be useful for this purpose. Using skos:hasTopConcept will trick SKOS-consuming applications into displaying a number of disconnected "concept hierarchies", which don't really represent the AAT hierarchy. Picking the roots of "concept hierarchies" at random depths in the general AAT hierarchy feels awkward. There are 887 such hierarchies, the majority of which (633) are no hierarchies at all:

- concepts: 38146
- top concepts: 887 (2.3% of all concepts)
- top concepts without children: 633 (71% of the top concepts)

So we finally decided to omit skos:topConceptOf and skos:hasTopConcept altogether. In lieu of this and to allow RDF crawlers to get all of AAT, we declare all Facets as void:rootResource of the AAT dataset (see Dynamic Descriptive Properties)

2.4 Sort Order

Usually, GVP Subjects and Terms are sorted alphabetically (for Subjects, by gvp:prefLabelGVP). But in some cases, a specific ("forced") ordering is set, e.g. for many subjects in the Periods and Styles facet.

Facets and Hierarchies don't have a useful ordering at present, so we ignore it. Regarding Terms and the other Subject Types:

- skos:OrderedCollection can be used to order the children of a GuideTerms
- SKOS does not have a standard way to order the children of a Concept, nor xl:Label (terms)

To accommodate a maximum number of consumers, we map this feature in 3 ways (belt, suspenders, AND a piece of string!)

1. With a custom property gvp:displayOrder (xsd:positiveInteger). You can use that with "order by" in SPARQL
2. OWLIM preserves the order of nodes as they are **first** inserted in the repository, and returns them in the same order in result sets. We have been careful to load the Subjects and Terms in the desired order, so you should also get them in this order, if you don't specify an "order by" clause. Note: there is no guarantee of this behavior, and no W3C standard that mandates it.
3. Using skos:OrderedCollection and iso:ThesaurusArray for GuideTerms and Concepts (see the next section)

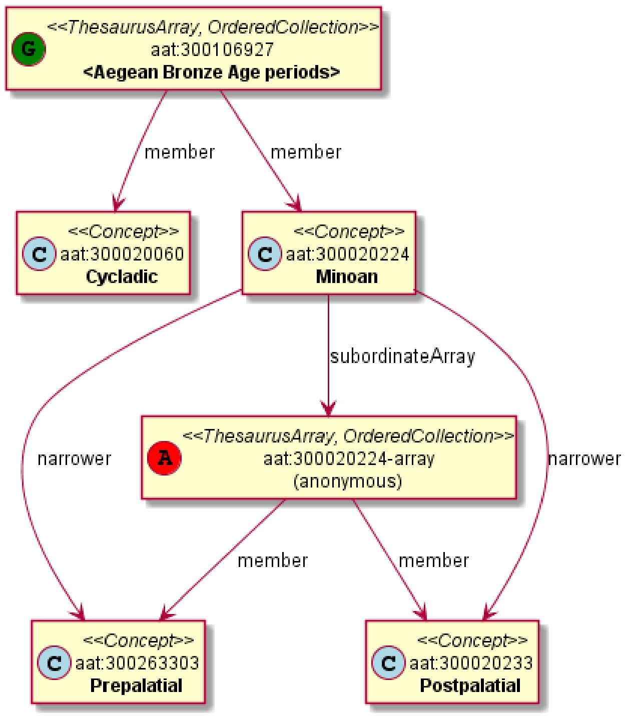
2.4.1 Sorting with Thesaurus Array

skos:OrderedCollection defines a standard way to order its children. In addition to skos:member, this uses skos:memberList, being an rdf:List (see section SKOS member vs memberList for details). iso:ThesaurusArray borrows the same. We illustrate the representation using two examples from the Periods and Styles facet:

- GuideTerm 300106927 <Aegean Bronze Age periods> and Concept 300020224 "Minoan"
- (Note: at present these don't have forced sort order, so instead explore 300018774 <Siberian periods>)

2.4.1.1 skos:member Structure

- The Guide Term (G) is represented as an iso:ThesaurusArray with skosxl:Label "<Aegean Bronze Age periods>"
- The Concept (C) "Minoan" is represented as skos:Concept. But it also has an iso:subordinateArray (A), that is an iso:ThesaurusArray, is *anonymous*, and serves only to hold the skos:OrderedCollection. *The anonymous array should not be displayed as a level in the hierarchy.*



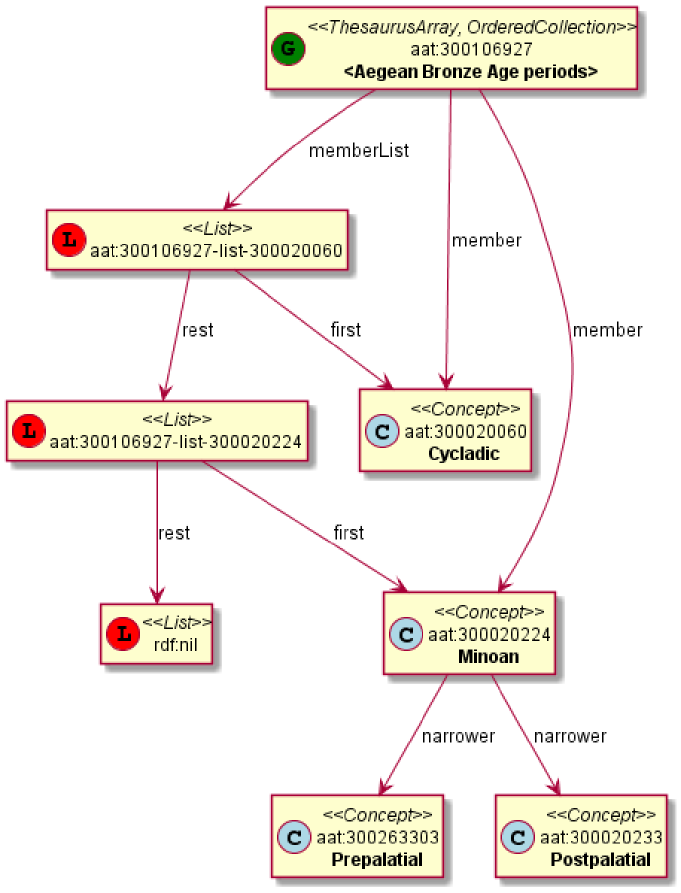
2.4.1.2 skos:memberList Structure

While the skos:member links establish collection membership (used both with unordered skos:Collection and ordered skos:OrderedCollection), additional skos:memberList structure provides the ordering of the members.

This uses the standard rdf:List construct. People often use blank nodes for rdf:List elements (and there is a special Turtle shortcut for this), but we use explicit URIs since this way it's easier to thread the list with the tooling used (R2RML).

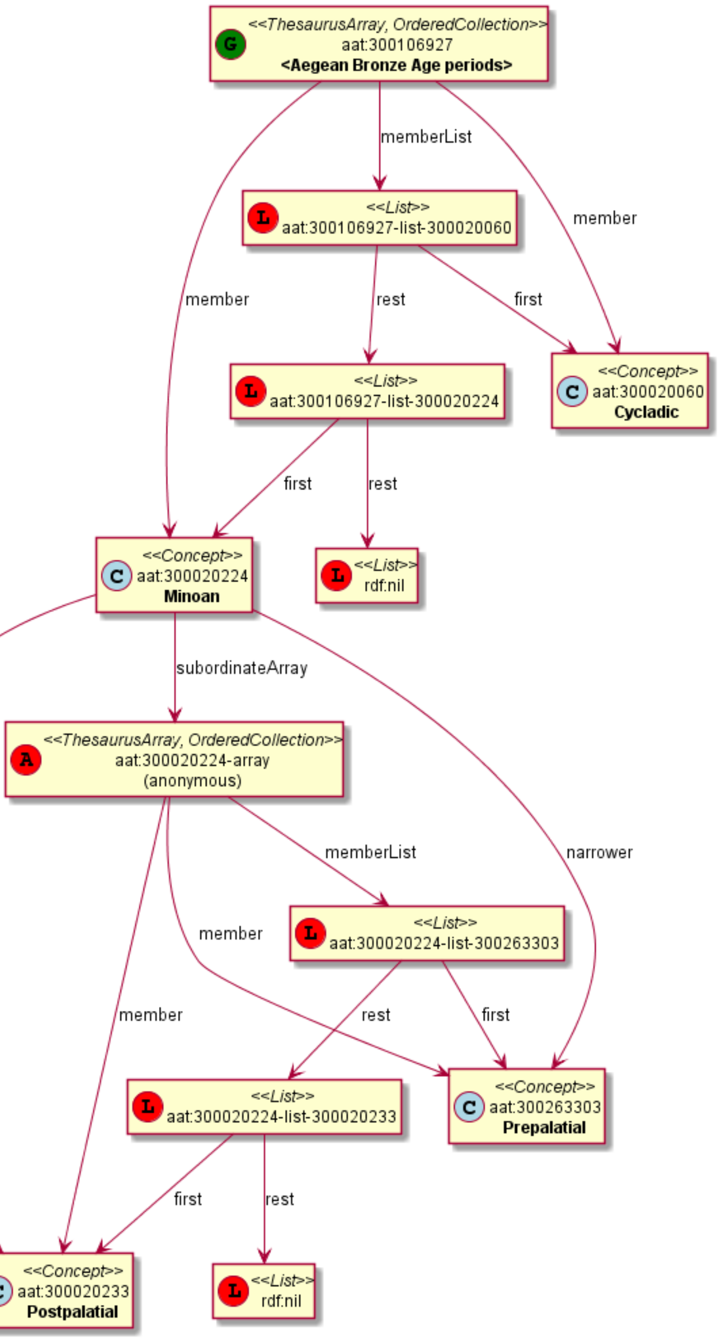
The list URIs have the form **aat:{p}-list-{c}** where {p} is the parent ID (owner of the list) and {c} is the child ID corresponding to the current list element.

For pedagogical purposes, we first show only the list of (G) <Aegean Bronze Age periods>. See the next section for the list of (C) "Minoan" (i.e. the full representation).



2.4.1.3 Full Representation

Adding the skos:memberList of (C) "Minoan", we get the full representation (not for the faint of heart!)


G <<ThesaurusArray, OrderedCollection>> aat:300106927 <Aegean Bronze Age periods>
memberList
member
member
L <<List>> aat:300106927-list-300020060
rest
first
L <<List>> aat:300106927-list-300020224
C <<Concept>> aat:300020060 Cycladic
first
rest
C <<Concept>> aat:300020224 Minoan
L <<List>> rdf:nil
subordinateArray
A <<ThesaurusArray, OrderedCollection>> aat:300020224-array (anonymous)
memberList
narrower
narrower
member
L <<List>> aat:300020224-list-300263303
member
rest
first
L <<List>> aat:300020224-list-300020233
C <<Concept>> aat:300263303 Prepalatial
first
rest
C <<Concept>> aat:300020233 Postpalatial
L <<List>> rdf:nil

2.5 Associative Relationships

GVP defines a plethora of associative relations between AAT subjects.

- Relations come in pairs of forward-inverse relation; symmetric relations are self-inverses.
- Every relation **instance** also has an inverse instance. i.e. if "A rel B" then there is relation "B inv(rel) A".
- We declare pairs as **owl:inverseOf** (and symmetric relations as owl:SymmetricProperty), but this won't infer new statements

2.5.1 Relationships Table

The Relationships Table (in PDF) describes the associative relationships used for AAT. If you'd like it in Excel format, please contact us. Legend:

- fcode=code of forward relation, icode=code of inverse relation
- domain=source (of forward relation), range=target (of forward relation); i.e. the kind of Concepts that it can connect
- frel=forward relation name, irel=inverse relation name,
- fdef=forward relation definition, idef=inverse relation definition,
- fexample=example of forward relation, iexample=example of inverse relation.

2.5.2 Relationship Cross-Walk

The Relationship Cross-Walk (matrix) (PDF) shows applicability of relations to kinds of Concepts (domain/range). To use:

- Begin with the vertical column on the left,
- Find intersection of Concept in vertical row with another Concept in horizontal column,
- See what relations are applicable

2.5.3 Relationship Representation

Almost the complete info from the Relationships Table is represented in RDF.

- Property URLs are generated as gvp:aat<code>_<name> where spaces in the name are replaced with "_".
- The properties are declared as sub-properties of skos:related. (This was changed recently because there were a few relation instances that connected non-Concepts)
- rdfs:domain and rdfs:range are declared as skos:Concept, whereas the "application" domain & range are emitted as comments only.
- <code> is emitted as dc:identifier
- "<name> - <range>" is emitted as dc:title
- Examples are emitted as multiple skos:example. We think these add a lot of clarity to the meaning of relations.
- dct:description includes <domain> - <name> - <range> and the examples

```
gvp:aat2208_locus-setting_for a owl:ObjectProperty;
  rdfs:subPropertyOf skos:related;
  rdfs:domain skos:Concept; rdfs:range skos:Concept;
  # domain "locus/setting"; range "things";
  dc:identifier "2208";
  skos:prefLabel "aat2208_locus-setting_for";
  dc:title "locus/setting for - things";
  skos:example "glassworks (buildings) are the locus/setting for glassware",
    "caves are the locus/setting for cave paintings" ;
  dct:description """locus/setting - [is] locus/setting for - things.
Example: glassworks (buildings) are the locus/setting for glassware;
  caves are the locus/setting for cave paintings""" .
```

```
gvp:aat2209_use-located_in a owl:ObjectProperty;
  rdfs:subPropertyOf skos:related;
  rdfs:domain skos:Concept; rdfs:range skos:Concept;
  # domain "things"; range "locus/setting";
  dc:identifier "2209";
  skos:prefLabel "aat2209_use-located_in";
  dc:title "use/located in - locus/setting";
  skos:example "glassware is used/located in glassworks (buildings)",
    "cave paintings are located in caves" ;
  dct:description """things -used/located in -  locus/setting.
Example: glassware is used/located in glassworks (buildings); cave paintings are located in caves""" .
gvp:aat2208_locus-setting_for owl:inverseOf gvp:aat2209_use-located_in.
  # or owl:SymmetricProperty if self-inverse
```

The dc:title allows you to construct nice displays. E.g. aat:300025419 "rope-makers" has a relation gvp:aat2292_work-live_in to "roperies". So its display can include:

Label:	rope-makers
Relations: *work/live in - locus/setting*	roperies

2.6 Obsolete Subject

GVP subjects may become obsolete as a result of editorial actions:

- Set as non-publishable (which basically means "deleted")
- Merged to another subject. They are mentioned by ID and name as "Recessive" in the "merge" Revision History action of that "Dominant" subject.

Obsolete subjects (more specifically Concepts) may have been used in client data. So in order not to leave such data hanging, we publish minimal information about them:

- **skos:prefLabel**: only the GVP preferred label (usually in English)
- **gvp:enfDate**: when it was obsoleted
- **dct:isReplacedBy**: merged to which subject

Currently obsolete subjects are 4.4% of valid subjects, which shows a good rate of editorial actions, and the importance of this information. Examples:

```
aat:300123456 a gvp:ObsoleteSubject; # Was made non-publishable
  skos:prefLabel "Made up subject";
  schema:endDate "2012-12-31T12:34:56"^^xsd:dateTime.

aat:300386746 a gvp:ObsoleteSubject; # Was merged to a dominant Subject
  skos:prefLabel "Buncheong";
  dct:isReplacedBy aat:300018699; # Punch'ong
  schema:endDate "2012-12-31T12:34:56"^^xsd:dateTime.
```

We want the inverse relation from the Dominant to the Recessive subject (dct:replaces). DCT doesn't have such declaration, so we add it:

```
dct:isReplacedBy owl:inverseOf dct:replaces.
```

Some would say this is "namespace hijacking". We call it adding info that's missing from DCT.

2.7 Language

GRI has gathered information about a plethora of Languages (some 1800), both ancient and modern. Other cultural heritage institutions have asked GRI to standardize in this area. Although there are several sources of language information (Lingvo, ISO, etc), a lot of the GRI languages are not defined there, e.g.:

- Liturgical Greek
- Chinese (transliterated Pinyin without tones)

GVP maintains language data in AAT, as concepts under 300389738 <languages and writing systems by specific example>.

- This data will be used uniformly across AAT, TGN and ULAN
- A mapping to IANA language tags is under development. We have covered all languages used in AAT (about 110) and are proceeding with languages used in TGN (about 100).

See the query Languages and ISO Codes to get this data.

2.7.1 IANA Language Tags

RDF literals use language tags as defined in the IANA Language Subtag Registry. Its structure (described in BCP47 sec 3.1) is not easy to read. So we wrote a script iana-lang-tags.pl that gets the registry, parses it, and writes it to a tab-delimited file. Open this file in excel and search to your heart's content.

The registry includes almost 9000 registrations (broken down by Type and Scope):

- 7769 language
- 227 extlang, eg ar-auz (Uzbeki Arabic)
- 116 language collection, e.g. bh (Bihari languages)
- 62 macrolanguages, e.g. zh (Chinese), cr (Cree)
- 4 special languages, e.g. und (Undetermined)
- 162 scripts, eg Latn (Latin), Japn (Japanese)
- 301 regions, e.g. US (United States), 021 (Northern America)
- 61 variants
- 67 redundant
- 26 grandfathered

2.7.2 GVP Language Tags

Despite the richness of IANA tags, we had to define new tags, using several extension mechanisms:

- Private language, e.g.
  - **x-byzantin**-Latn for Byzantine Greek (transliterated)
  - **x-khasian** for Khasian
- Private language used in specific region, e.g.
  - **qqq-002** for African language
  - **qqq-142** for Asian language
  - **qqq-ET** for Ethiopian (not specified which: Boro/Borna, Karo, Male...)
- Private modifier, e.g.
  - grc-Latn-**x-liturgic** for Liturgical Greek
  - ber-Latn-**x-dialect** for Berber Dialects (transliterated)
  - fa-Latn-**x-middle** for Persian, Middle (transliterated)
  - zh-Latn-pinyin-**x-notone** for Chinese (transliterated Pinyin without tones)
  - **x-frisian**. IANA/ISO has codes for its predecessor Old Frisian and its dialects West, Saterland and North Frisian, but not for Frisian itself.

GVP has a language 300389645 "Undetermined" (gvp_lang:und). It's used infrequently in AAT, but is prevalent in TGN (1468740 out of 1843907, or 78% of the terms have undetermined language). IANA has a corresponding tag **und** "Undetermined", but we decided NOT to emit it: due to the Open World Assumption, unknown/undetermined info does not need to be emitted. Instead of the info on the left, we emit the one on the right:

```
aat:300312114
  skos:altLabel "bukağı"@und;
  skosxl:altLabel aat_term:1000408720-und.
aat_term:1000408720-und
  skosxl:literalForm "bukağı"@und;
  gvp:term "bukağı"@und;
  dct:language aat:300389645, gvp_lang:und.
```

```
aat:300312114
  skos:altLabel "bukağı";
  skosxl:altLabel aat_term:1000408720.
aat_term:1000408720
  skosxl:literalForm "bukağı" ;
  gvp:term "bukağı".
```

2.7.3 Language Tag Case

Term URLs have the language tag appended. But you may have noticed a discrepancy in the case used to spell them, e.g.

```
aat_term:1000024386-en-US xl:literalForm "currycombs"@en-us .
```

This is not an error:

- RFC 5646 (BCP47) section 2.1.1 Formatting of Language Tags demands "At all times, language tags and their subtags, including private use and extensions, are to be treated as case insensitive".
- The SPARQL function langMatches() treats them case-insensitively

Nevertheless, it's an unpleasant discrepancy, since RFC 5646 continues "consistent formatting and presentation of language tags will aid users. The format of subtags in the registry is RECOMMENDED as the form to use in language tags." In other words, RFC 5646 demands that implementations are case-insensitive, and recommends that they are case-preserving.

Conventionally, language tags are written in lower-case except:

- Script is capitalized, e.g. el-**Latn**
- Region is in uppercase, e.g. en-**US**

You should use langMatches() to compare language tags (see Find Terms by Language Tag), or spell the tag in the exact case used in the repository (e.g. el-latn and en-us).

We undertook some steps to get RDF vendors to normalize case as recommended by RFC 5646, or at least preserve it:

- Posted bug against Perl's RDF::Trine::Node::Literal (Sep 2013), which is used in the R2RML tool that we use (RDF2RDB)
- Posted lang_normalize routine that can be reused by other vendors as well
- Perl fix adopted and implemented on Github (Jan 2014)
- Posted bug against Sesame (model and RIO): SES-1999, SES-1659 (Jan 2014), which is still open
- Proposed change to the:
  - RDF 1.1 standard: "Lexical representations of language tags MAY be normalized, according to BCP47 section 2.1.1. "Formatting of Language Tags" (country codes in upper case, script codes capitalized, the rest in lower case). Language tags MAY also be normalized by converting all to lower case, but BCP47 normalization is preferred"
  - SPARQL lang() function: "lang() MAY normalize the language tag as described in RDF 1.1 Concepts and Abstract Syntax sec 3.3 Literals. It is recommended that lang() normalizes the literal according to BCP47 section 2.1.1, and not by converting it all to lower case."

Jena appears to store the lang tag as provided, which is better than storing as lowercase

2.7.4 Language Tags and Sources

Language data includes names in several languages, ISO language codes, IANA language tags. E.g. see http://vocab.getty.edu/aat/300389115:

- Language names (skos:prefLabel): "Portuguese (language)"@en, "portugais"@fr, "Portugiesisch"@de
- Language codes/tags (skos:altLabel): "por"@en, "pt"@en

You can find out where the tags come from by exploring the term's Source:

- The skos:altLabel "**pt**"@en corresponds to the skosxl:altLabel aat_term:1000576998-en that has a dct:source that dct:isPartOf aat_source:2000075479, which is "ISO 639-1 Alpha-2 codes for names of languages"

The skos:altLabel "**por**"@en corresponds to the skosxl:altLabel aat_term:1000576997-en that has dct:source aat_source:2000075493, which is "ISO 639-2 Alpha-3 codes for names of languages".

Notes:

- Previously we had an intermediate bibo:DocumentPart node that was dct:isPartOf aat_source:2000075479 (respectively aat_source:2000075493), with a constant bibo:locator saying "ISO 2-character". We considered this a parasitic node, so we have now removed it
- "**pt**" and "**por**" are not English words, so using @en for them is (strictly speaking) not correct. In the future we may introduce special lang tags @x-iso6392, @x-iso6393, @x-iana to mark the language tags with.

2.7.5 Language Dual URLs

AAT languages with assigned language tag have dual URLs, e.g. for Portuguese:

- aat:300389115: "systemic" URL of the language as a concept in the Languages hierarchy.
- gvp_lang:pt: "logical" URL that's used as dct:language in Terms (skosxl:prefLabel, skosxl:altLabel) and Notes (skos:scopeNote). Coincidentally, the URLs of Terms include the language tag, e.g.

```
aat_term:1000011071-pt
  skosxl:literalForm "asbestos"@pt;
  dct:language gvp_lang:pt.
```

The dual language URLs are declared owl:sameAs, e.g. http://vocab.getty.edu/aat/300389115.ttl has this triple:

```
aat:300389115 owl:sameAs gvp_lang:pt.
```

You can access the language data using either URL. We include a query to find all Language URLs.

Why didn't we use established language URLs, e.g. from Lexvo? (For example, the Lingvoj site has started using Lexvo URLs in 2010, in a spirit of reuse and cooperation). We would be glad to, but many of the GVP languages are specific so we had to add custom tags. See GVP Language Tags for examples.

2.8 Term

GVP includes multilingual terms and rich information about them, including preferred status, sources, contributors, revision history, etc. We use SKOSXL (skosxl:Label) for the full information, and plain SKOS (literal labels) to allow SKOS-only clients to access the literals (see SKOS and SKOS-XL). Please note that in AAT each Term is owned by exactly one Subject (i.e. is dependent on the subject), whereas SKOS-XL potentially allows one xl:Label to be reused between Concepts (eventually in different roles)

Terms carry the following information:

- **dc:identifier**: numeric ID, also used in the term URL. See Identifiers
- **gvp:term**: the proper (own) label, e.g. "rhea"
- **gvp:qualifier**: serves to clarify and disambiguate terms with the same spelling but different meaning, e.g. "vessels" vs "species"
- **skosxl:literalForm**: concatenation of term and parenthesized qualifier, e.g.
  - "rhea (vessels)" meaning "rhyta" (a kind of drinking vessel) vs
  - "rhea (species)" meaning "Boehmeria nivea" (Chinese grass)
- **dct:language** see Language. If the language is gvp_lang:<lang>, then the URL of the Term is aat_term:<identifier>-<lang>, and the language tag of the previous 3 properties is @<lang>
- **gvp:displayOrder**, see Sort Order
- Historic Information about historic applicability of the term
- Links to Source
  - Can be to a global source (bibo:Document), or to a local source (bibo:DocumentPart)
  - Can be plain dct:source; or sub-properties thereof (gvp:sourcePreferred, gvp:sourceNonPreferred, gvp:sourceAlternatePreferred), specifying whether the term is Preferred, NonPreferred or AlternatePreferred for the source
- Links to Contributor.
  - Similar to Source links, these can be plain dct:contributor, or sub-properties thereof (gvp:contributorPreferred, gvp:contributorNonPreferred, gvp:contributorAlternatePreferred)

2.8.1 Term Characteristics

Terms have a number of enumerated characteristics. A lot of these are optional, i.e. could be missing.

- **gvp:termDisplay**: Use for Display (where natural order is preferred) or Indexing/lists (where inverted order is appropriate)?
- **gvp:termFlag**: Vernacular, Loan Term
- **gvp:termKind**: Abbreviation, Common term, Chemical Name, Full term, Jargon or Slang, Neologism, Scientific or Technical term
- **gvp:termPOS**: Noun, Plural Noun, Singular Noun, Both Singular and Plural, Past Participle, Verbal Noun/Gerund, Adjectival
- **gvp:termType**: Descriptor, Alternate Descriptor, Used for Term

These are captured as Concepts in small subsidiary ConceptSchemes, with their own definitions and examples. You can see these with the query Ontology Values.

Note: in the first release of AAT, the value URLs were under http://vocab.getty.edu/aat. But we have now moved them one level up, in preparation for adding TGN.

2.9 Scope Note

All Concepts (but not all non-concept Subjects) have definitions (scope notes) in the basic GVP languages: English, Spanish and Dutch (Chinese is upcoming). Notes are represented as skosxl:Label and have similar (but simpler) information like Terms:

- **dc:identifier**: numeric ID, also used in the URL. See Identifiers
- **skosxl:literalForm**: the note itself
- **dct:language** see Language.
- **gvp:displayOrder**, see Sort Order
- Historic Information about historic applicability
- Links to Source
- Links to Contributor.

The SKOS Primer section 4.2 Advanced Documentation Features provides an example of representing rich notes. It uses rdf:value to hold the literal, and doesn't define the node's type. We preferred to use explicit type (skosxl:Label) and to structure the information same as for Terms.

2.10 Identifiers

We map the database ID's of Subjects, Terms, Scope Notes, Sources and Contributors to dc:identifier. These are random numeric codes that are also used in the GVP URLs of these entities.

- They are guaranteed to stay permanent, so you can use them in your own data (you'll probably only want to use the Subject URLs aat:{m}).
- If a Subject is merged to another, we emit it as Obsolete Subject

2.11 Notations

A notation is a code or number used to uniquely identify a concept within the scope of a given concept scheme. Unlike Terms, notations are not normally recognizable as a sequence of words in any natural language. DDC, UDC, STW and other well-known thesauri use notations. Example codes include "T58.5" or "303.4833".

We use notations for the following:

- Traditional GVP Facet/Hierarchy Codes, e.g.

```
aat:300241490 skos:notation "V.R". # Components (Hierarchy Name)
```

  - Note: on the GVP website, lower-level subjects (Guide Terms and Concepts) duplicate the same code as the higher level, so we don't include a notation for those

- Short character codes of Term Characteristics

```
<http://vocab.getty.edu/aat/term/kind/ScientificOrTechnical> skos:notation "S".
```

2.12 Source

GVP tracks sources of Terms (labels), Subjects (concepts), and Scope Notes. Sources may be catalogs, encyclopedias, other publications, web sites, databases, etc.

We represent sources as bibo:Document with the following info:

- **dc:identifier** (see Identifiers)
- **bibo:shortTitle**: brief title
- **dct:title**: full title
- **skos:note**: bibliographic note

For example:

```
aat_source:2000030301 a bibo:Document;
  dc:identifier "2000030301";
  bibo:shortTitle "Chenhall, Revised Nomenclature (1988)";
  dct:title "Chenhall, Robert G. The Revised Nomenclature for Museum Cataloging: ... ".
aat_source:2000051089 a bibo:Document;
  dc:identifier "2000051089"
  bibo:shortTitle "AATA database (2002-)";
  dct:title "Getty Conservation Institute (GCI). database of AATA Online... 2002-. ".
aat_source:2000052946 a bibo:Document;
  dc:identifier "2000052946";
  bibo:shortTitle "Encyclopedia Britannica Online (2002-)";
  dct:title "Encyclopædia Britannica. ... http://www.eb.com/ (1 July 2002).".
```

Sources are applied to Subjects and Scope Notes using dct:source.

For terms, GVP may record whether the term is Preferred, NonPreferred or AlternatePreferred for the source, so we use dct:source or sub-properties thereof: gvp:sourcePreferred, gvp:sourceNonPreferred, gvp:sourceAlternatePreferred.

The multipart URLs like aat_source:2000051089-term-1000198841 are explained in the next section:

```
# terms of "rhyta" in English, Greek, Spanish
aat_term:1000198841-en
  gvp:sourceNonPreferred aat_source:2000049728;
  dct:source aat_source:2000051089-term-1000198841.
aat_term:1000198841-el-Latn
  gvp:sourceNonPreferred aat_source:2000049728;
  dct:source aat_source:2000051089-term-1000198841.
aat_term:1000198841-es
  gvp:sourceNonPreferred aat_source:2000049728;
  dct:source aat_source:2000051089-term-1000198841.

# subject 300198841 (rhyta)
aat:300198841
  dct:source aat_source:2000030301-subject-300198841;
  dct:source aat_source:2000052378.

# notes in English and Dutch
aat_scopeNote:34904
  dct:source aat_source:2000046502.
aat_scopeNote:83378
  dct:source aat_source:2000051213.
```

2.12.1 Local Sources

When applied, a source may carry a locator. To attach this info, we need an intermediate node (Local Source) represented as bibo:DocumentPart:

- **dct:isPartOf**: points to the global (original) source

- **bibo:locator**: a page number, heading, database ID, or other accession information

The URL of local sources consists of the original source URI, followed by the ID of the thing being described (term, subject or note), e.g.:

```
aat_source:2000051089-term-1000198841 a bibo:DocumentPart;
  dct:isPartOf aat_source:2000051089;
  bibo:locator "128257 checked 26 January 2012".
aat_source:2000030301-subject-300198841 a bibo:DocumentPart;
  dct:isPartOf aat_source:2000030301;
  bibo:locator "horn, drinking".
```

Note: bibo:locator is defined as "A description (often numeric) that locates an item within a containing document or collection", which matches our usage.

Its domain is declared as bibo:Document. Consequently our bibo:DocumentParts are also inferred to be bibo:Document. Although unintended, this is ok, since the two classes are not disjoint (we think this is an omission in BIBO: it should allow bibo:locator to "locate a DocumentPart within its containing Document" as well).

2.13 Contributor

GVP tracks contributors of Terms (labels), Subjects (concepts), and Scope Notes

We represent Contributors as foaf:Agent with the following info:

- **dc:identifier** (see Identifiers)
- **foaf:nick**: abbreviation (e.g. CDBP-DIBAM)
- **foaf:name**: full name (e.g. Centro de Documentación de Bienes Patrimoniales (Dirección de Bibliotecas, Archivos y Museos; Santiago, Chile)

For example:

```
aat_contrib:10000000 a foaf:Agent;
  dc:identifier "10000000";
  foaf:nick "VP";
  foaf:name "Getty Vocabulary Program".
aat_contrib:10000131 a foaf:Agent;
  dc:identifier "10000131";
  foaf:nick "CDBP-DIBAM";
  foaf:name "Centro de Documentación de Bienes Patrimoniales …".
aat_contrib:10000205 a foaf:Agent;
  dc:identifier "10000205";
  foaf:nick "Bureau AAT";
  foaf:name "Bureau AAT, RKD (Netherlands Institute for Art History; …)".
```

Contributors are attached to Subjects and Notes using dct:contributor.

For terms, GVP may record whether the term is Preferred, NonPreferred or AlternatePreferred for the contributor, so we use dct:contributor or sub-properties thereof: gvp:contributorPreferred, gvp:contributorNonPreferred, gvp:contributorAlternatePreferred (similar to Sources), e.g.:

```
# term "rhyta"
aat_term:1000198841-en
  gvp:contributorNonPreferred aat_contrib:10000131;
  gvp:contributorPreferred aat_contrib:10000000.
aat_term:1000198841-el-Latn
  gvp:contributorNonPreferred aat_contrib:10000131;
  gvp:contributorPreferred aat_contrib:10000000.
aat_term:1000198841-es
  gvp:contributorNonPreferred aat_contrib:10000131;
  gvp:contributorPreferred aat_contrib:10000000.
```

```
# subject "rhyta"
aat:300198841
  dct:contributor aat_contrib:10000131;
  dct:contributor aat_contrib:10000000.

# notes in English and Dutch
aat_scopeNote:34904
  dct:contributor aat_contrib:10000000.
aat_scopeNote:83378
  dct:contributor aat_contrib:10000205.
```

2.14 Historic Information

GVP includes historic information about the validity of terms, hierarchical relations and associative relations. We use the following properties:

- **gvp:historicFlag**: with values Current, Historic, Both (or undetermined)
- **schema:startDate, schema:endDate**: Previously we used custom sub-properties of dct:valid (defined as "Date (often a range) of validity of a resource"). But some found it confusing to have two values for dct:valid, so we preferred to use schema: properties (which don't infer dct:valid).
  - Literals are emitted in proper XSD date format and carry a type according to its precision, e.g.: "1940"^^xsd:gYear, "1940-06"^^xsd:gMonthYear, "1940-06-15"^^xsd:date
  - Years are spelled with least 4 digits. Years BC are expressed as negative, eg: "0100"^^xsd:gYear, "0010"^^xsd:gYear, "-0100"^^xsd:gYear, "-10000"^^xsd:gYear
- **rdfs:comment**: note on historic applicability.

2.14.1 Applying to Terms

Applying historic information to Terms represented as explicit nodes (skosxl:Label) is straightforward:

```
aat_term:1000002693-en
  a skosxl:Label;
  skosxl:literalForm "lambruscatura"@en ;
  gvp:historicFlag <http://vocab.getty.edu/historic/historic> ;
  schema:startDate "0900"^^xsd:gYear ;
  schema:endDate "1700"^^xsd:gYear ;
  rdfs:comment "Medieval term for wainscoting".
```

2.14.2 Applying to Relations

To apply historic information to relations, we use the RDF Reification vocabulary. It allows to represent a relation instance (i.e. triple) as an rdf:Statement, and use rdf:subject, rdf:object and rdf:predicate to address its components.

We give these statements explicit URLs, using the aat_rel: prefix, the IDs of the related subjects, and the relation name: "broader" for hierarchical, and the specific aatNNNN_* for associative.

```
aat_rel:300107346-broader-300020541
  a rdf:Statement;
  rdf:subject aat:300107346; # Early Imperial
  rdf:predicate gvp:broaderPreferred;
  rdf:object aat:300020541; # Imperial (Roman)
  rdfs:comment "ca. 27 BCE-68 CE";
  schema:startDate "-0017"^^xsd:gYear;
  schema:endDate "0068"^^xsd:gYear .
```

```
aat_rel:300020271-aat2812_followed-300020269
  a rdf:Statement;
  rdf:subject aat:300020271; # Second Dynasty (Egyptian)
  rdf:predicate gvp:aat2812_followed;
  rdf:object aat:300020269; # First Dynasty (Egyptian)
  rdfs:comment "Second Dynasty began ca. 2775 BCE";
  schema:startDate "-2785"^^xsd:gYear;
  schema:endDate "-2765"^^xsd:gYear.
```

- In the case of hierarchical relations, historic information is asserted against gvp:broaderPreferred or gvp:broaderNonPreferred and is **not copied** to inferred relations.
- In the case of associative relations, the same historic information is asserted against both the forward and inverse relations (e.g. aat2812 and aat2811) with the roles of rdf:subject and rdf:object swapped. For a symmetric relation (e.g. aat2203_associated_with), the same info is asserted twice, with the roles of rdf:subject and rdf:object swapped.

2.15 Revision History

GVP keeps extensive info about actions on Subjects and Sources: over 600k records as of Feb 2014. Although extensive, there is no guarantee the info is comprehensive, especially for very old actions, and for Publish events (issued).

The various kinds of actions are listed below, together with approximate numbers (see Count Revision Actions). Actions on sub-entities of Subjects (terms, notes, associative relations) are emitted for the Subject.

what	dc:type	rdf:type	count	dc:description
Subject	created	prov:Create	45798	
Subject	updated	prov:Modify	105082	
Subject	term added	prov:Modify	293212	<term> (<term.id>)
Subject	term deleted	prov:Modify	15501	<term> (<term.id>) OR was <term>
Subject	note created	prov:Modify	10626	Language: <lang.name> (<lang.id>)
Subject	note updated	prov:Modify	20719	Language: <lang.name> (<lang.id>)
Subject	moved	prov:Modify	21588	Old Parent: <parent.term> (<parent.id>)
Subject	parent added	prov:Modify	3410	<parent.term> (<parent.id>)
Subject	relation added	prov:Modify	33321	<this.term> (<this.id>) 'relation.name' <related.term> (<related.id>)
Subject	relation deleted	prov:Modify	12512	<this.term> (<this.id>) 'relation.name' <related.term> (<related.id>)
Subject	merged	prov:Modify	soon	Dominant: <this.term> (<this.id>), Recessive: <rec.term> (<rec.id>)
Subject	issued	prov:Publish	2432	To be published to Website and LOD within 2 weeks
Source	created	prov:Create	35974	
Source	updated	prov:Modify	16306	
Source	merged	prov:Modify	423	Dominant: <this.name>, Recessive: <rec.name>

There are over 10M "term updated" actions that are not very interesting and are therefore elided.

2.15.1 Revision History Representation

We map revision actions to PROV-O, using the PROV-DC mapping for inspiration. We use classes and properties from prov-o.ttl and prov-dc-refinements.ttl. But because PROV has a high level of complexity (see the examples in section PROV), we use a rather simplified mapping.

Revision records use the aat_rev: (respectively aat_source_rev:) prefix and have the following data:

- **rdf:type**: prov:Activity, and a specific type as listed in the table above: prov:Create, prov:Modify, or prov:Publish (these come from the PROV-DC refinements)
- **dc:type**: a literal as listed in the table above

- **prov:startedAtTime**: timestamp of the action (xsd:dateTime)
- **dc:description**: additional narrative about the action, such as which Recessive subject was merged, or what is the language of the note that was added.

Links between the entity and its actions:

- **skos:changeNote**: from entity to action. The SKOS Advanced Documentation pattern shows this for skos:Concept. For uniformity, we use the same property for any gvp:Subject, and also for Sources (foaf:Agents). The property doesn't define a domain, so that's permissible.
- **prov:wasGeneratedBy**: from entity to the prov:Create action
- **prov:used**: from prov:Modify or prov:Publish to the entity

Timestamps on the entity: we emit the action timestamps also as timestamps on the entity, using DCT properties:

- prov:Create → dct:created
- prov:Modify → dct:modified: there is one for each Modify action. If you prefer to have only the latest timestamp, let us know
- prov:Publish → dct:issued

2.15.2 Revision History for Subject

Let's say subject 300018699 was created by action 12345, modified by action 12346 (a term was added), and issued (published) by action 12347. We map it thus:

```
aat:300018699
  skos:changeNote aat_rev:12345, aat_rev:12346, aat_rev:12347;
  prov:wasGeneratedBy aat_rev:12345;
  dct:created  "2014-01-02T01:02:03"^^xsd:dateTime;
  dct:modified "2014-01-03T01:02:03"^^xsd:dateTime;
  dct:issued   "2014-01-04T01:02:03"^^xsd:dateTime.
aat_rev:12345 a prov:Activity, prov:Create;
  dc:type "created";
  prov:startedAtTime "2014-01-02T01:02:03"^^xsd:dateTime.
aat_rev:12346 a prov:Activity, prov:Modify;
  prov:used aat:300018699;
  dc:type "term added";
  dc:description "leggings, puttee (1000248060)";
  prov:startedAtTime "2014-01-03T01:02:03"^^xsd:dateTime.
aat_rev:12347 a prov:Activity, prov:Publish;
  prov:used aat:300018699;
  dc:type "issued";
  prov:startedAtTime "2014-01-04T01:02:03"^^xsd:dateTime.
```

2.15.3 Revision History for Source

Let's say source 2000053040 was created by action 12345 and modified by action 12346 (another source was merged); there are no Publishing actions for sources. (Note: these are different from the Subject actions described above, even if they have the same ID)

```
aat_source:2000053040
  skos:changeNote aat_source_rev:12345, aat_source_rev:12346;
  prov:wasGeneratedBy aat_source_rev:12345;
  dct:created  "2014-01-02T01:02:03"^^xsd:dateTime;
  dct:modified "2014-01-03T01:02:03"^^xsd:dateTime.
aat_source_rev:12345 a prov:Activity, prov:Create;
  dc:type "created";
  prov:startedAtTime "2014-01-02T01:02:03"^^xsd:dateTime.
aat_source_rev:12346 a prov:Activity, prov:Modify;
  prov:used aat_source:2000053040;
  dc:type "merged";
  dc:description "Recessive: Magness, Archaeology of Qumran … (2003) (2000076344)";
  prov:startedAtTime "2014-01-03T01:02:03"^^xsd:dateTime.
```

3 Additional Features

3.1 Inference

An important decision is what sort of inference level to use in the GVP SPARQL endpoint. Considerations:

- Advantage: the more powerful the inferencing, the fewer facts need to be stated explicitly and the easier it will be to develop and maintain the conversion.
- Disadvantage: users that download the files and don't have the same or higher level of inference will be at a disadvantage

We use an approach that uses the advantage while avoiding the disadvantage:

- The conversion process produces only basic facts: Explicit Exports
- Then we use OWLIM's powerful inference features (RDFS, OWL RL, OWL QL) and INSERT queries to infer all required consequences and materialize them in the repository.
- We extract from OWLIM export files with all inferred facts: Per-Entity Exports and Total Exports

In this way you can use the Total Exports without need for additional inference, or use the Explicit Exports and provide the inferences described below.

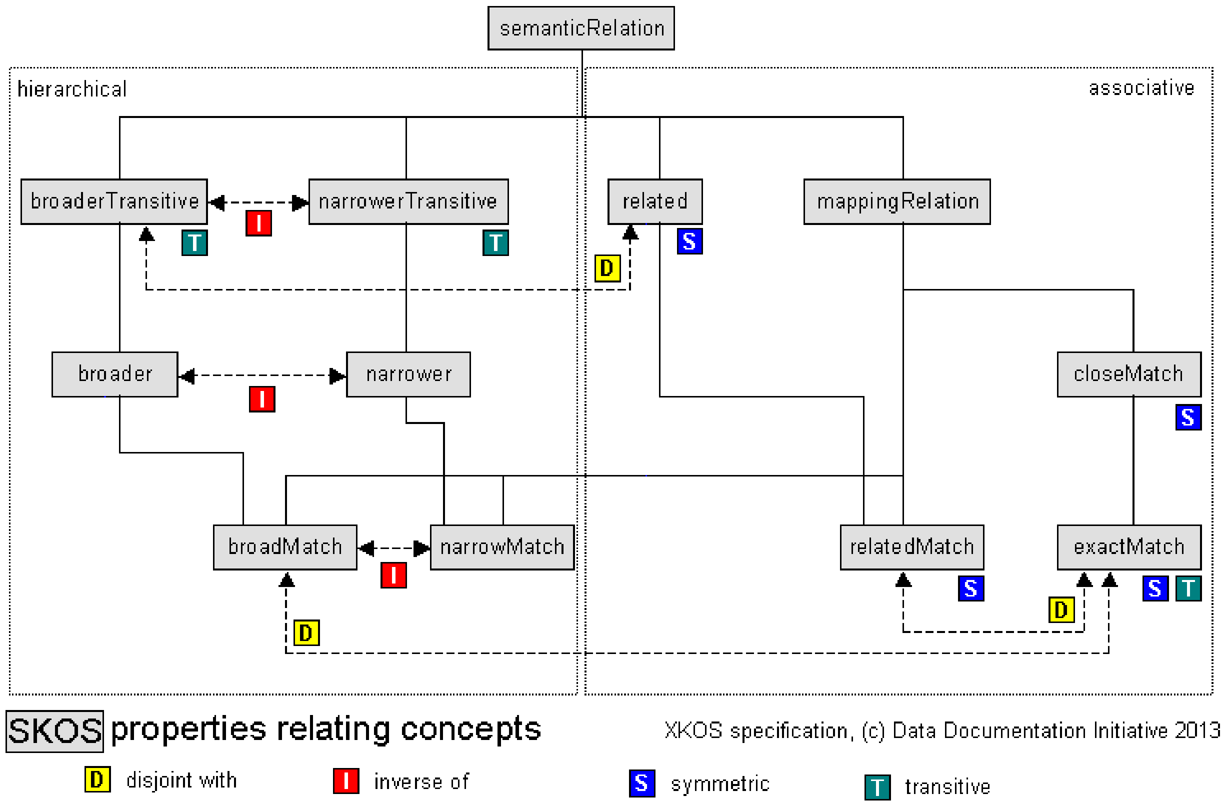
We also run INSERT queries to generate Dynamic Descriptive Properties, but you don't need to do that: you can get the VOID file directly.

3.1.1 SKOS Inference

The SKOS property hierarchy is shown below, together with an indication of property characteristics: S=symmetric, I=inverse, T=transitive, D=disjoint:

```
skos:semanticRelation          rdfs:label
  skos:related (S)               skos:prefLabel
    skos:relatedMatch (S)        skos:altLabel
  skos:broaderTransitive (T)     skos:hiddenLabel
    skos:broader (I)
      skos:broadMatch (I)      skos:note
  skos:narrowerTransitive (IT)   skos:changeNote
    skos:narrower (I)            skos:definition
      skos:narrowMatch (I)       skos:editorialNote
  skos:mappingRelation           skos:example
    skos:closeMatch (S)          skos:historyNote
      skos:exactMatch (ST)       skos:scopeNote
    skos:relatedMatch (S)
    skos:broadMatch (I)
    skos:narrowMatch (I)
```

The following diagram shows a bit more information:



The definitions of SKOS properties dictate the inferences shown above. The following constructs are used in the SKOS ontology:

- **rdfs:subPropertyOf**, e.g. to infer broader → broaderTransitive
- **owl:TransitiveProperty**: to make the transitive closure, e.g. of broaderTransitive
- **owl:SymmetricProperty**: to infer "A related B" → "B related A"
- **owl:inverseOf**: to infer e.g. broader → narrower and vice versa

3.1.2 SKOS member vs memberList

The SKOS Reference, sec 9. "Concept Collections" has a semantic constraint (S36) that's relevant to Sorting with Thesaurus Array. S36 requires that every item in a skos:memberList should also have a direct link skos:member. The Open Annotation specification, sec 4.3 "List" suggests a way to infer the direct links from the list using owl:propertyChainAxiom.

However, we have chosen to emit explicit skos:member links, because we do that for unordered collections anyway, and it would not have been easier to do differently for ordered collections. So this inference rule is not needed.

3.1.3 SKOS-XL Inference

The SKOS recommendation requires every **skosxl:Label** to also be present as a simple SKOS literal label (see Dumbing-Down SKOSXL labels to SKOS lexical labels). This requirement S55 is not formally stated in the SKOS-XL ontology, but can be implemented in two ways

SKOS-XL Property Chains

We state the following OWL Property Chains. (Note: we must use property chains for BTG, BTP, BTI Inference, so we also use them for this inference.)

```
skos:prefLabel   owl:propertyChainAxiom (skosxl:prefLabel   skosxl:literalForm).
skos:altLabel    owl:propertyChainAxiom (skosxl:altLabel    skosxl:literalForm).
skos:hiddenLabel owl:propertyChainAxiom (skosxl:hiddenLabel skosxl:literalForm).
```

SKOS-XL Insert Queries

If you don't have an OWL RL compliant repository/reasoner, you can implement the required inference by running the following INSERT queries after loading the Explicit Exports files:

```
insert {?x skos:prefLabel   ?z} where {?x skosxl:prefLabel   ?y. ?y skosxl:literalForm ?z};
insert {?x skos:altLabel    ?z} where {?x skosxl:altLabel    ?y. ?y skosxl:literalForm ?z};
insert {?x skos:hiddenLabel ?z} where {?x skosxl:hiddenabel  ?y. ?y skosxl:literalForm ?z};
```

3.1.4 BTG, BTP, BTI Inference

To infer "Extended" versions (meaningful closures) of the GVP Hierarchical Relations, we use the rules defined in the paper Compositionality of ISO 25964 Hierarchical Relations (V.Alexiev, J.Lindenthal, A.Isaac, May 2014, submitted). We summarize the rules in the following table.

- Each cell expresses a chain of two relations, and whether they can infer a third one
- The row gives the **left** relation, the column gives the **right** relation, and the cell gives the **conclusion**
- BT*x means "BT* or BT*E". In is a reference to the particular inference, used in the next section
- For each positive conclusion, we give an example; and for some negative we give a counter-example

The inference can be expressed in 3 equivalent ways:

- Using SPARQL property path notation (as in the paper)

```
left/right => conclusion
```

- Using N3 Rules:

```
{?x left ?y. ?y right ?z} => {?x conclusion ?y}
```

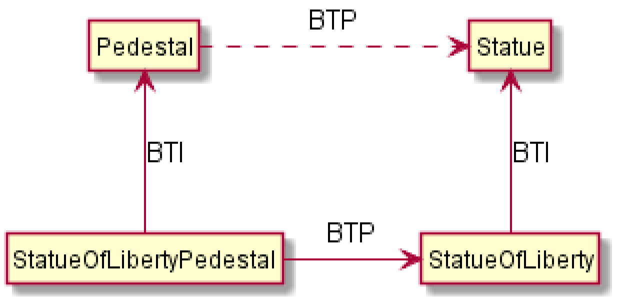
- !Using OWL property chains (as implemented in the next section)

```
conclusion owl:propertyChainAxiom (left right)
```

left/right	**BTGx**	**BTPx**	**BTIx**
BTGx	I2: BTGE: numerous examples	I3: BTP: *beak irons* BTG *anvil components* BTP <*anvils and anvil accessories*>	I4: no
BTPx	I5: BTPE: *anvil components* BTP <*anvils and anvil accessories*> BTG <*forging and metal-shaping tools*>	I6: BTP: *Sofia* BTP *Bulgaria* BTP *Europe*	I7: no: *Sofia* BTP *Bulgaria* BTI *country*, but *Sofia* is no *country*
BTIx	I8: BTIE: *Mt Athos* BTI *orthodox religious center* BTG *Christian religious center*	I9: no, see below	I10: no

I9: Consider the example *Statue of Liberty pedestal* BTI *pedestals* BTP *statues*. The particular *Statue of Liberty pedestal* has no relation to *statues* in general (is neither an instance nor a part of *statues*). In the figure below we could infer the dashed relation (generalization of BTP from one instance thereof), i.e.

```
{?x BTI ?y. ?x BTP ?z. ?z BTI ?t} => {?y BTP ?t}
```



But in I9 we have only 3 nodes in sequence, not 4 nodes.

3.1.5 BTG, BTP, BTI Axioms

For these inferences we cannot use a finite number of INSERTs because the property chains involved are recursive. We need to use rule-based inference. OWLIM easily makes such inferences using the OWL RL ruleset (but the default Horst ruleset is faster).

First we feed BT* to BT*E:

```
# I1: BTG->BTGE, BTP->BTPE, BTI->BTIE: basic inferences
gvp:broaderGeneric    rdfs:subPropertyOf gvp:broaderGenericExtended.
gvp:broaderPartitive  rdfs:subPropertyOf gvp:broaderPartitiveExtended.
gvp:broaderInstantial rdfs:subPropertyOf gvp:broaderInstantialExtended.
```

These inferences are explained in the previous section:

```
# I2: BTGx/BTGx->BTGE
gvp:broaderGenericExtended owl:propertyChainAxiom
  (gvp:broaderGenericExtended gvp:broaderGenericExtended).

# I3: BTGx/BTPx->BTPE
gvp:broaderPartitiveExtended owl:propertyChainAxiom
  (gvp:broaderGenericExtended gvp:broaderPartitiveExtended).
# I5: BTPx/BTGx->BTPE
gvp:broaderPartitiveExtended owl:propertyChainAxiom
 (gvp:broaderPartitiveExtended gvp:broaderGenericExtended).
# I6: BTPx/BTPx->BTPE
gvp:broaderPartitiveExtended owl:propertyChainAxiom
  (gvp:broaderPartitiveExtended gvp:broaderPartitiveExtended).

# I8: BTIx/BTGx->BTIE
gvp:broaderInstantialExtended owl:propertyChainAxiom
  (gvp:broaderInstantialExtended gvp:broaderGenericExtended).
```

This defines gvp:broaderExtended (BTE) as a disjunction of BTGE, BTPE, BTIE:

```
# I11: BTGE|BTPE|BTIE->BTE
gvp:broaderGenericExtended    rdfs:subPropertyOf gvp:broaderExtended.
gvp:broaderPartitiveExtended  rdfs:subPropertyOf gvp:broaderExtended.
gvp:broaderInstantialExtended rdfs:subPropertyOf gvp:broaderExtended.
```

3.1.6 broaderPreferredExtended Rules

We define gvp:broaderPreferredExtended as:

- Meaningful closure (appropriate extension) of gvp:broaderPreferred, or equivalently as
- Specialization of gvp:broaderExtended along broaderPreferred only.

This can be expressed in N3 Rules as:

```
{?x gvp:broaderPreferred ?y}
   => {?x gvp:broaderPreferredExtended ?y}
{?x gvp:broaderPreferredExtended ?y. ?y gvp:broaderPreferred ?z. ?x gvp:broaderExtended ?z}
   => {?x gvp:broaderPreferredExtended ?z}
```

The first rule is a trivial rdfs:subPropertyOf. But the second rule involves:

- Conjunction, so it cannot be implemented with OWL axioms.
- Recursion, so it cannot be implemented with SPARQL INSERT (nor SPIN Rules), unless one is willing to run INSERTS many times, until nothing new is inferred.

We implement it using OWLIM Rules (which do full forward chaining):

```
Prefices {
  gvp : http://vocab.getty.edu/ontology#
}
Rules
{
Id: broaderPreferredExtended
  x <gvp:broaderPreferredExtended> y
  y <gvp:broaderPreferred> z
  x <gvp:broaderExtended> ?z
  ----------
  x <gvp:broaderPreferredExtended> z
}
```

If you want to implement this inference yourself and you don't have OWLIM, you can do it at the expense of introducing an auxiliary property gvp:broaderPreferredTransitive (therefore more triples), e.g.:

```
# Axioms
gvp:broaderPreferred rdfs:subPropertyOf gvp:broaderPreferredTransitive.
gvp:broaderPreferredTransitive a owl:TransitiveProperty.
# Insert Query
insert {?x gvp:broaderPreferredExtended ?y}
where {?x gvp:broaderPreferredTransitive ?y. ?x gvp:broaderExtended ?y}
```

3.1.7 ISO Insert Queries

We infer SKOS and ISO Standard Hierarchical Relations from GVP Hierarchical Relations as follows:

- skos:member and iso:subordinateArray, to implement the Hierarchy Structure and Sorting with Thesaurus Array
- skos:broader, skos:narrower that "thread" the skos:Concept hierarchy as a subset of the complete gvp:Subject hierarchy.
- ISO BTG, BTP, BTI from GVP BTGE, BTPE, BTIE (the "Extended" relations implemented in the previous section), but only when the "Extended" relation connects two Concepts directly

This inference involves complex property conditions. OWL axioms cannot infer properties under conjunction nor negation, and OWLIM rules cannot infer properties under negation.

But the rules are not recursive, so we implement the inference with INSERT queries after loading the Explicit Exports files:

```
# Q1: add skos:member below each iso:ThesaurusArray
insert {?x skos:member ?y}
where {?x a iso:ThesaurusArray; gvp:narrower ?y};

# Q2: add iso:subordinateArray from skos:Concept to iso:ThesaurusArray
insert {?x iso:subordinateArray ?y}
where {?x a skos:Concept; gvp:narrower ?y. ?y a iso:ThesaurusArray};

# Q3: add skos:broader (and its inverse skos:narrower) threading the skos:Concept hierarchy
insert {?x skos:broader ?y}
where {?x a skos:Concept. ?y a skos:Concept. ?x gvp:broaderExtended ?y
  filter not exists {
    ?z a skos:Concept.
    ?x gvp:broaderExtended ?z. ?z gvp:broaderExtended ?y}};
```

```
# Q4: add ISO BTG, BTP, BTI from GVP BTGE, BTPE, BTIE connecting pairs of Concepts directly.
# We use the fact that Q3 has asserted skos:broader for such pairs.
insert {?x iso:broaderGeneric ?y}
  where {?x skos:broader ?y;  gvp:broaderGenericExtended ?y};
insert {?x iso:broaderPartitive ?y}
  where {?x skos:broader ?y; gvp:broaderPartitiveExtended ?y};
insert {?x iso:broaderInstantial ?y}
  where {?x skos:broader ?y; gvp:broaderInstantialExtended ?y};
```

The query below is not used anymore, since we decided not to emit Top Concept indication:

```
# add skos:topConceptOf for those Concepts having no broader Concept in the same scheme
insert {?x skos:topConceptOf ?scheme }
where {?x a skos:Concept; skos:inScheme ?scheme
  filter not exists {?y a skos:Concept; skos:inScheme ?scheme; gvp:narrower ?x}}
```

3.2 Alignment

A key potential benefit of LOD is the ability to create and exploit linkages between datasets, e.g. alignments. AAT provides a few alignments, and GVP hopes that external contributors (such as VUA's Amalgame project) will provide more.

3.2.1 LCSH Alignment

We generate some (about 300) alignments to LCSH based on explicit mention of LCSH ID in bibo:locator of a Local Source:

- When the source dct:isPartOf aat_source:2000046735 (LC Subject Authority Headings) and bibo:locator includes a number of >=8 digits, OR
- When bibo:locator consists of a prefix "sh", followed by a number of >=8 digits

This check is performed for sources at both Subject and Term level. The alignment is emitted for the AAT subject.

Take for example aat:300008736 "waterfalls" (see http://vocab.getty.edu/aat/300008736.ttl):

```
aat:300008736 a gvp:Concept ;
  gvp:prefLabelGVP aat_term:1000008736-en ;
  gvp:prefLabelLoC aat_term:1000008736-en .
aat_term:1000008736-en a skosxl:Label
  gvp:sourcePreferred aat_source:2000046735-term-1000008736 .
aat_source:2000046735-term-1000008736 a bibo:DocumentPart ;
  dct:isPartOf aat_source:2000046735 ;
  bibo:locator "sh 85145720" .
aat_source:2000046735 a bibo:Document ;
  bibo:shortTitle "LC Subject Authority Headings [online] (2002-)" .
```

It has a Term (prefLabel) that has a Local Source (bibo:DocumentPart) that is both part of LCSH, and has a bibo:locator matching the "sh" ID pattern. From this we infer the alignment:

```
aat:300008736
  skos:exactMatch <http://id.loc.gov/authorities/subjects/sh85145720>.
```

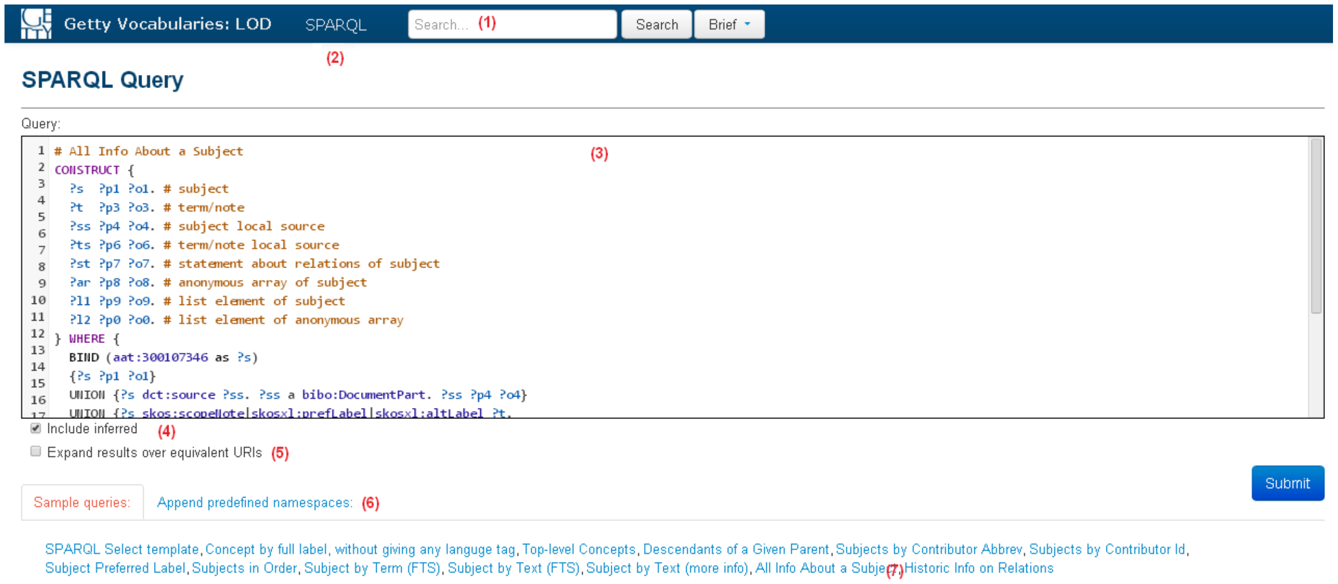
3.2.2 AATNed Alignment

AATNed is the project producing the Dutch translation of AAT. It started publishing LOD earlier than AAT, so some institutions (e.g. the Rijksmuseum) have already started using their URLs (e.g. http://service.aat-ned.nl/skos/300024521). AATNed has made the decision to merge into AAT, so the new GVP URLs should be used (e.g. http://vocab.getty.edu/aat/300024521). The IDs correspond, so there is no need to provide alignment links

```
 <http://service.aat-ned.nl/skos/300024521>
  dct:isReplacedBy <http://vocab.getty.edu/aat/300024521>
```

3.3 Forest UI

Forest is a UI framework for creating semantic applications by Ontotext. GVP uses a customized version of Forest that provides the following features (the red numbers on the screen-shots are explained in the text below).



SPARQL Query

Query:

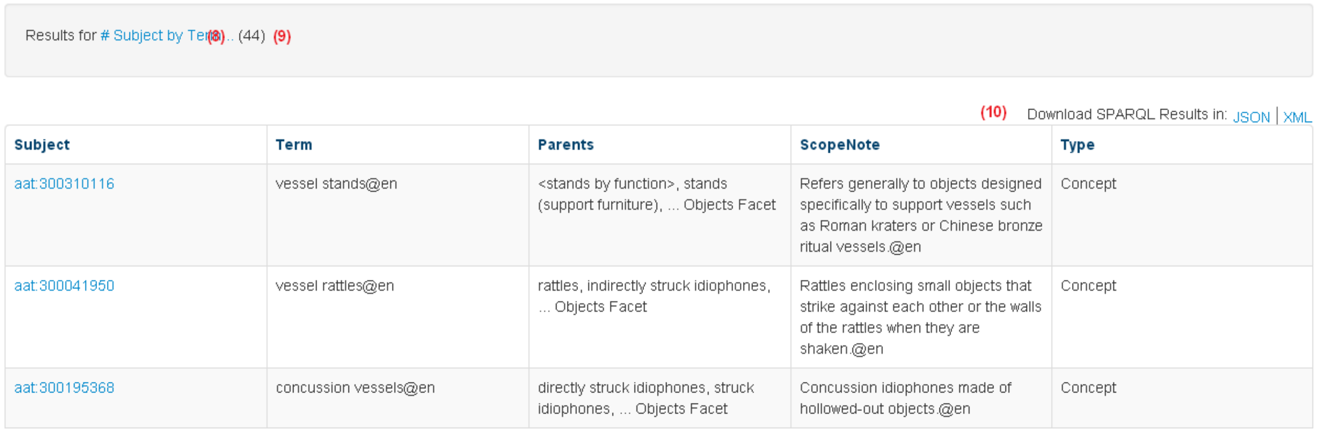
```
# All Info About a Subject                                         (3)
CONSTRUCT {
  ?s  ?p1 ?o1. # subject
  ?t  ?p3 ?o3. # term/note
  ?ss ?p4 ?o4. # subject local source
  ?ts ?p6 ?o6. # term/note local source
  ?st ?p7 ?o7. # statement about relations of subject
  ?ar ?p8 ?o8. # anonymous array of subject
  ?l1 ?p9 ?o9. # list element of subject
  ?l2 ?p0 ?o0. # list element of anonymous array
} WHERE {
  BIND (aat:300107346 as ?s)
  {?s ?p1 ?o1}
  UNION {?s dct:source ?ss. ?ss a bibo:DocumentPart. ?ss ?p4 ?o4}
  UNION {?s skos:scopeNote|skosxl:prefLabel|skosxl:altLabel ?t.
```

[✓ Include inferred] (4)

Expand results over equivalent URIs (5)

SPARQL Select template, Concept by full label, without giving any languge tag, Top-level Concepts, Descendants of a Given Parent, Subjects by Contributor Abbrev, Subjects by Contributor Id, Subject Preferred Label, Subjects in Order, Subject by Term (FTS), Subject by Text (FTS), Subject by Text (more info), All Info About a Subject (7) Historic Info on Relations

- Full Text Search (1)
- SPARQL query endpoint (2), supporting SPARQL 1.1
  - Accessible both through REST URL, and through an interactive form with syntax highlighting and auto-indent (3)
  - You can specify whether to return only Explicit triples, or also Inferred triples (4)
  - You can specify whether to expand results across owl:sameAs and return owl:sameAs assertions. (This applies to Language Dual URLs) (5)
  - All External Prefixes and GVP Prefixes used by the representation are predefined in the repository, so you don't need to add them. There is a function to append a predefined namespace (6), but you won't need to use it, except to examine these namespaces
- Sample Queries, accessible through links below the SPARQL query box (7)



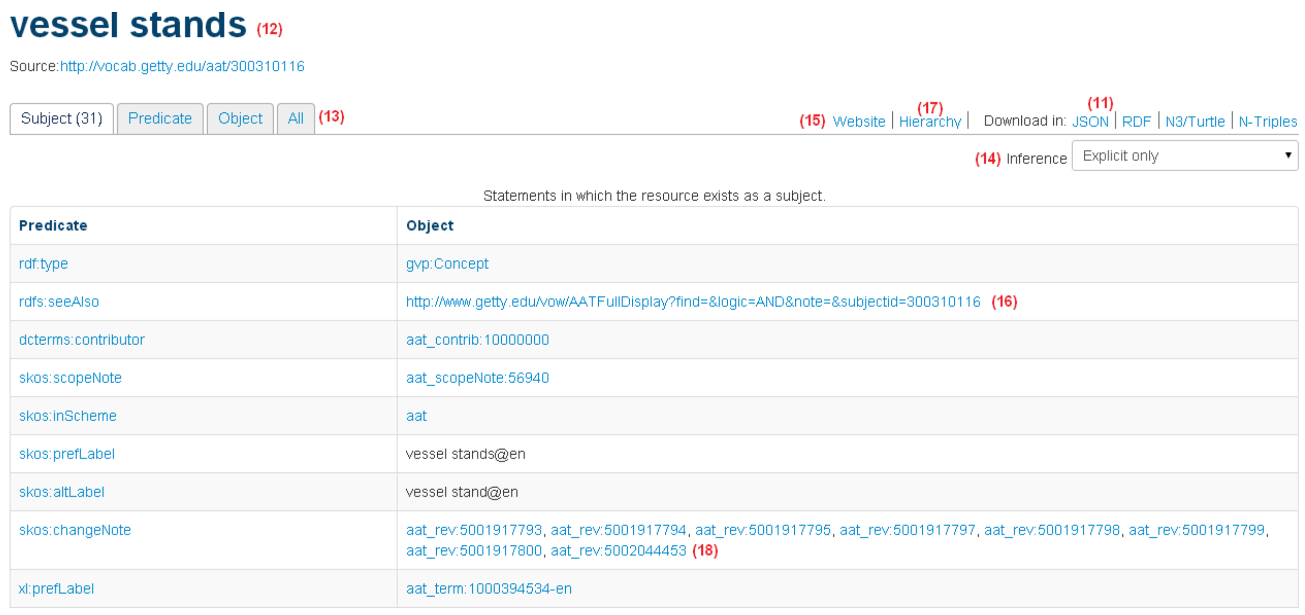
Results for # Subject by Term (8)... (44) (9)

(10) Download SPARQL Results in: JSON | XML

Subject	Term	Parents	ScopeNote	Type
aat:300310116	vessel stands@en	<stands by function>, stands (support furniture), ... Objects Facet	Refers generally to objects designed specifically to support vessels such as Roman kraters or Chinese bronze ritual vessels.@en	Concept
aat:300041950	vessel rattles@en	rattles, indirectly struck idiophones, ... Objects Facet	Rattles enclosing small objects that strike against each other or the walls of the rattles when they are shaken.@en	Concept
aat:300195368	concussion vessels@en	directly struck idiophones, struck idiophones, ... Objects Facet	Concussion idiophones made of hollowed-out objects.@en	Concept

- Editing of the previous query (8).
  Click on the query link "Results for…" and **not** on the SPARQL link in the header
- Number displayed and total number of results (9).

- Returning query results in HTML, RDF/XML, Turtle, NTriples, JSON (10).
  RDF/XML and Turtle are available only for CONSTRUCT queries
- Query result pagination (for HTML)
- Semantic Resolution of resources (GVP URLs), including content negotiation
-



vessel stands (12)

Source:http://vocab.getty.edu/aat/300310116

Subject (31) | Predicate | Object | All (13)

(15) Website | (17) Hierarchy | Download in: (11) JSON | RDF | N3/Turtle | N-Triples

(14) Inference: Explicit only

Statements in which the resource exists as a subject.

Predicate	Object
rdf:type	gvp:Concept
rdfs:seeAlso	http://www.getty.edu/vow/AATFullDisplay?find=&logic=AND&note=&subjectid=300310116 (16)
dcterms:contributor	aat_contrib:10000000
skos:scopeNote	aat_scopeNote:56940
skos:inScheme	aat
skos:prefLabel	vessel stands@en
skos:altLabel	vessel stand@en
skos:changeNote	aat_rev:5001917793, aat_rev:5001917794, aat_rev:5001917795, aat_rev:5001917797, aat_rev:5001917798, aat_rev:5001917799, aat_rev:5001917800, aat_rev:5002044453 (18)
xl:prefLabel	aat_term:1000394534-en

Resource representations in HTML, RDF/XML, Turtle, NTriples, JSON (11).

- Label of the resource. For subjects this is usually but not always the first prefLabel (12)
- Tabs to show triples where the resource plays different roles (subject, predicate, object) or all triples (13)
- Selector whether to include Explicit, Inferred or all triples (14)
- For the semantic formats (RDF/XML, Turtle, NTriples, JSON), all triples are returned
- For the independent entities (Subjects, Sources, Contributors), the semantic formats include all triples of all owned objects as well (see Per-Entity Exports)
- Link to the Subject page on GVP's website (15).
  This is the same link as rdfs:seeAlso (16) but it would take an extra click if you use that.
- A link to the GVP page showing the Subject's position in the Hierarchy (17)
- Conversely, the GVP site has links to download each of the semantic formats.
- Clickable links to explore all related resources (18)

3.4 Full Text Search

You can search for Subjects using the Full Text Search box. Two indexes are provided that can be selected with a drop-down:

- **Brief** (luc:term): includes all terms and subject ID.
- **Full** (luc:text): includes all terms, qualifiers, subject ID, and scope notes.

This search uses the Lucene FTS engine that is built into OWLIM.

The search results include the following columns: subject ID (with link), GVP-preferred term (gvp:prefLabelGVP), abbreviated parent string, abbreviated scope note, and subject type).

- Result pagination is provided
- All language representations are searched, but the results are returned always in English.

The following pre-processing of the query phrase is performed:

- Characters that are not letter/digit are replaced with a space.

- Words are wild-carded with *
- A conjunction (AND) is added between the words
- For Chinese hieroglyphs, no analysis is performed and the entered hieroglyphs must match exactly.

For programmatic querying, use the predicates luc:text and luc:term in SPARQL (see Full Text Search Query).

3.5 Descriptive Information

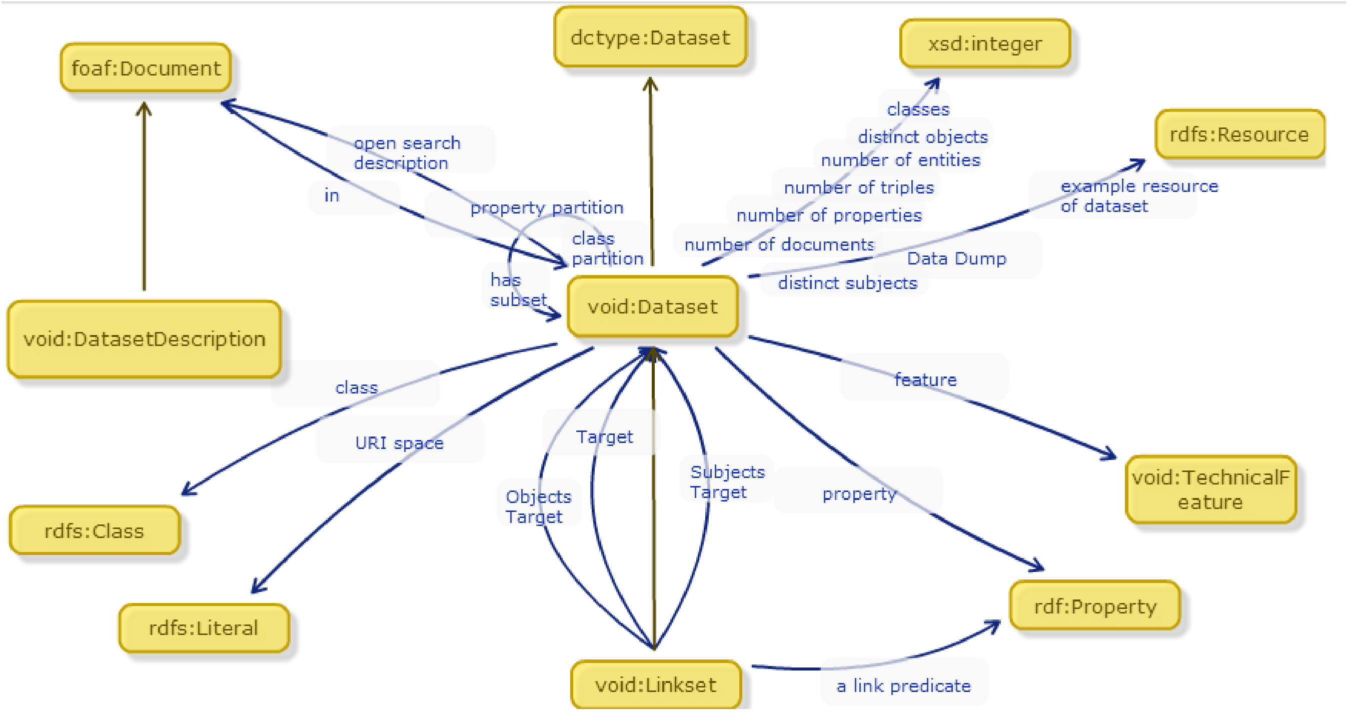
Machine-readable descriptive information is crucial to allow semantic agents to discover, register, crawl, analyze and summarize datasets. It provides the backbone of information for LOD registries such as the DataHub (http://datahub.io). Yet, the creators of the famous LOD cloud diagram (http://lod-cloud.net) report that many datasets are missing basic descriptive and licensing information, which makes it harder for people and agents to consume LOD.

We provide comprehensive info about the AAT dataset, ontology and concept scheme.

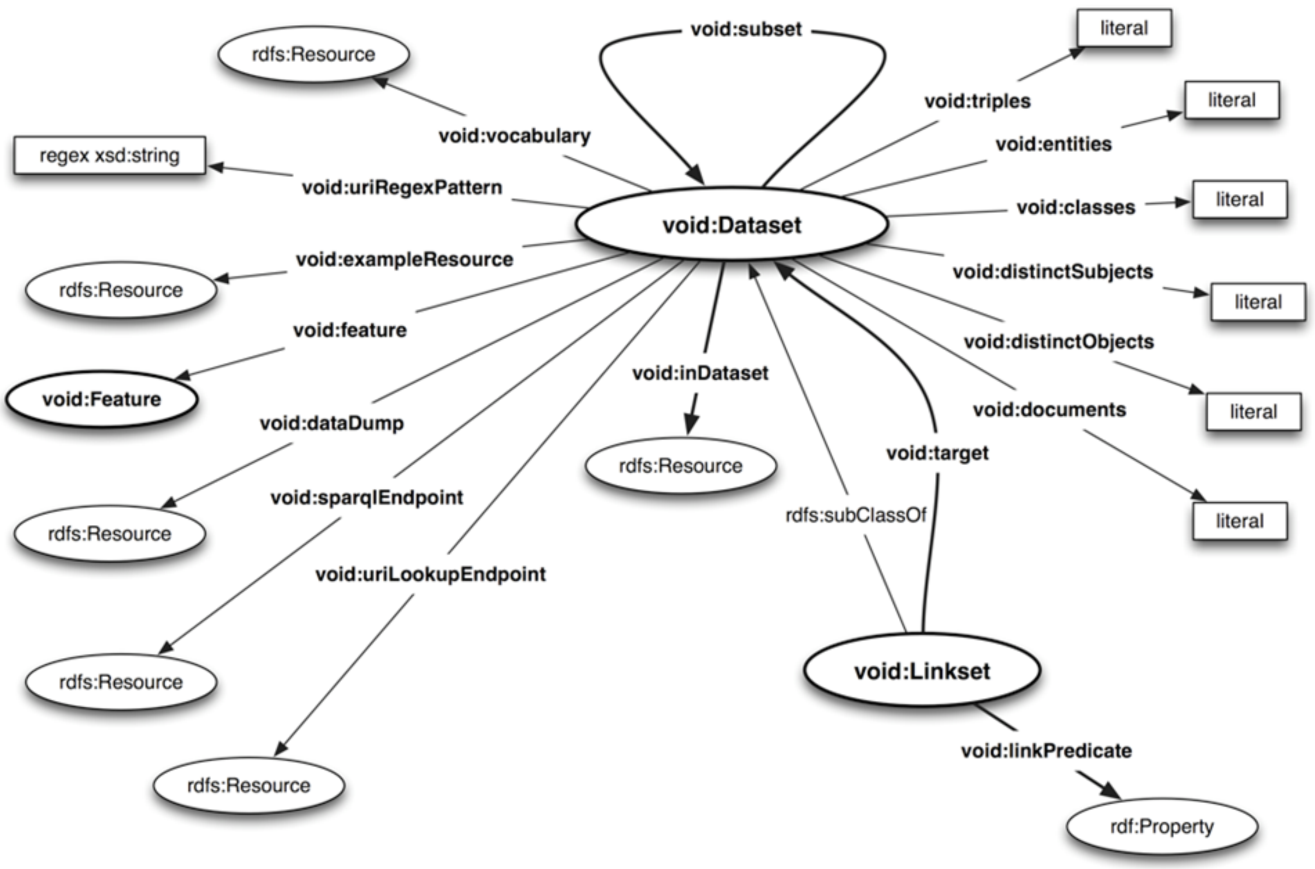
- We used this useful list of metadata-description vocabularies from LOV
- We use properties from most of the vocabularies listed in Descriptive Prefixes. Why so many? Because there is overlap between the different vocabularies, yet each has something extra to say
- The basic ontologies are VOID, DCAT (not to be confused with DCT!), ADMS, CC .They are described in the subsections below.
- We also use the ubiquitous DC, DCT; and a few properties from DCTYPE, VANN, VOAG, WDRS, WV.
- The AAT dataset is already registered at the DataHub: http://datahub.io/dataset/getty-aat
- We hope that our descriptive info fulfills the Guidelines for Collecting Metadata on Linked Datasets in the Data Hub, and intend to validate this with http://validator.lod-cloud.net/

3.5.1 VOID

Vocabulary of Interlinked Datasets (VOID) is the main ontology for describing RDF datasets. The VOID specification Describing Linked Datasets with the VoID Vocabulary was published by W3C on 3 March 2011. The VoID vocabulary definition (namespace document) provides a reference of all classes and properties, and the following domain model:



The slideshare presentation VoID: Metadata for RDF Datasets (Richard Cyganiak, May 14, 2012, p.13) provides a more lucid domain model:

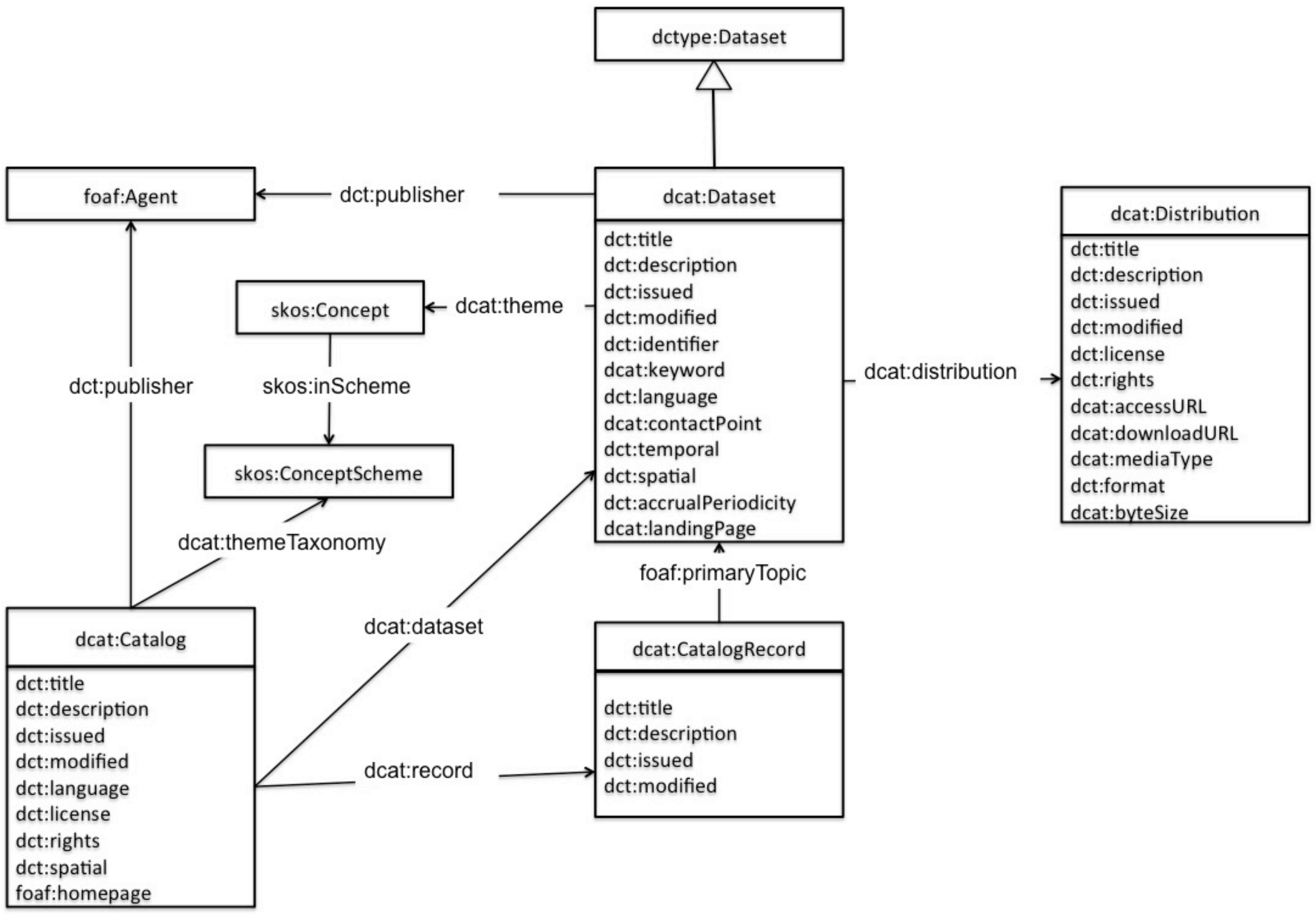


VOID covers a number of areas:

- Descriptive info (who, when, what) using DC, DCT
- Structural info, interlinking the dataset, its description, data dumps, SPARQL endpoint, used ontology, etc
- Access URLs and mechanisms, e.g. RDF dumps and SPARQL endpoint
- Vocabularies, properties and classes used
- Statistics about size (number of triples), including per property/class
- Resource URI patterns

3.5.2 DCAT

The Data Catalog Vocabulary (DCAT) is a W3C Recommendation published on 16 January 2014. It is used to describe datasets and data catalogs.

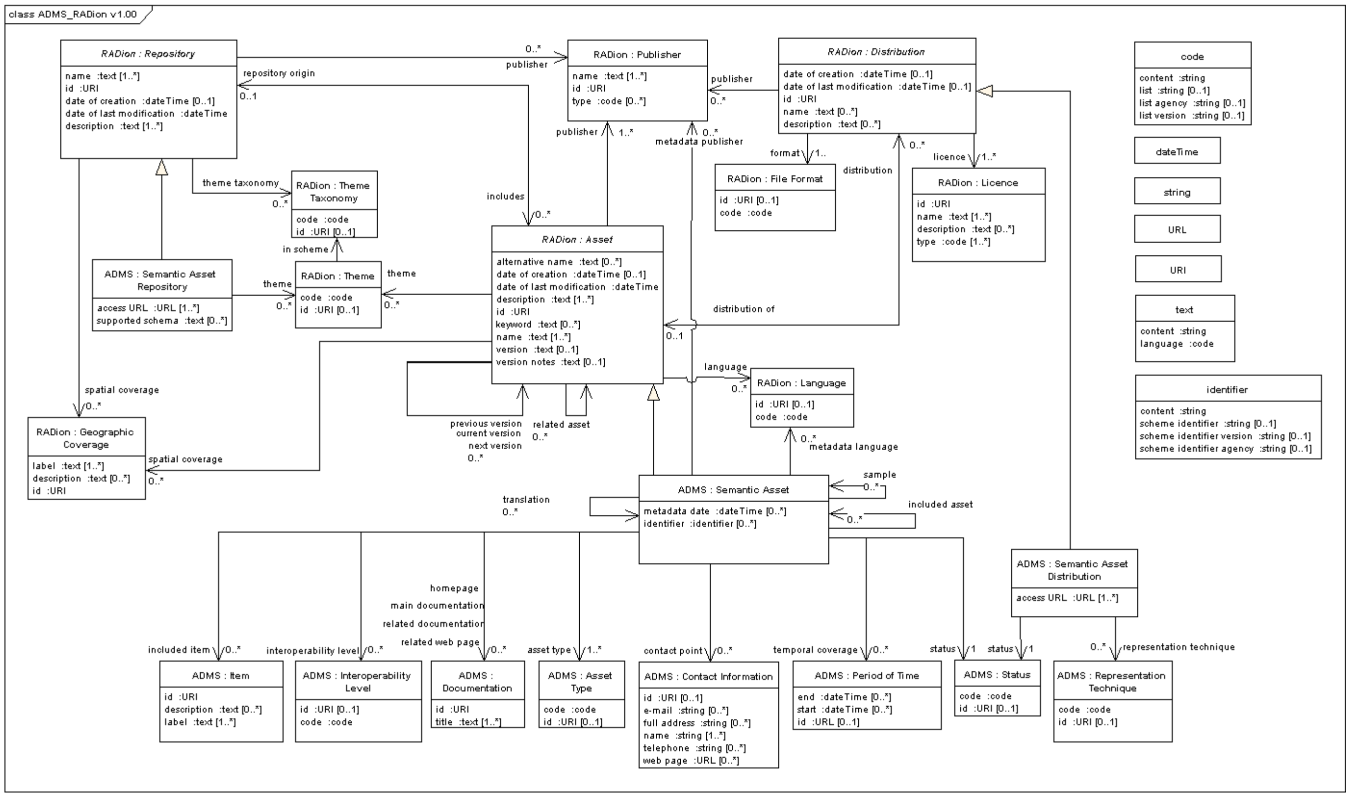


3.5.3 ADMS

The Asset Description Metadata Schema (ADMS) is a metadata vocabulary created by the Interoperability Solutions for European Public Administrations (ISA) Programme of the European Commission.

- While DCAT describes data sets, ADMS describes reusable metadata (e.g. xml schemata, generic data models) and reference data (e.g. code lists, taxonomies, dictionaries, vocabularies).
- While DCAT is focused on data catalogs, ADMS is focused on the assets within a catalog

Version 1.00 was released on 18 April 2012. The ADMS Conceptual Model is based on an earlier ontology called RADION and is fairly complex:



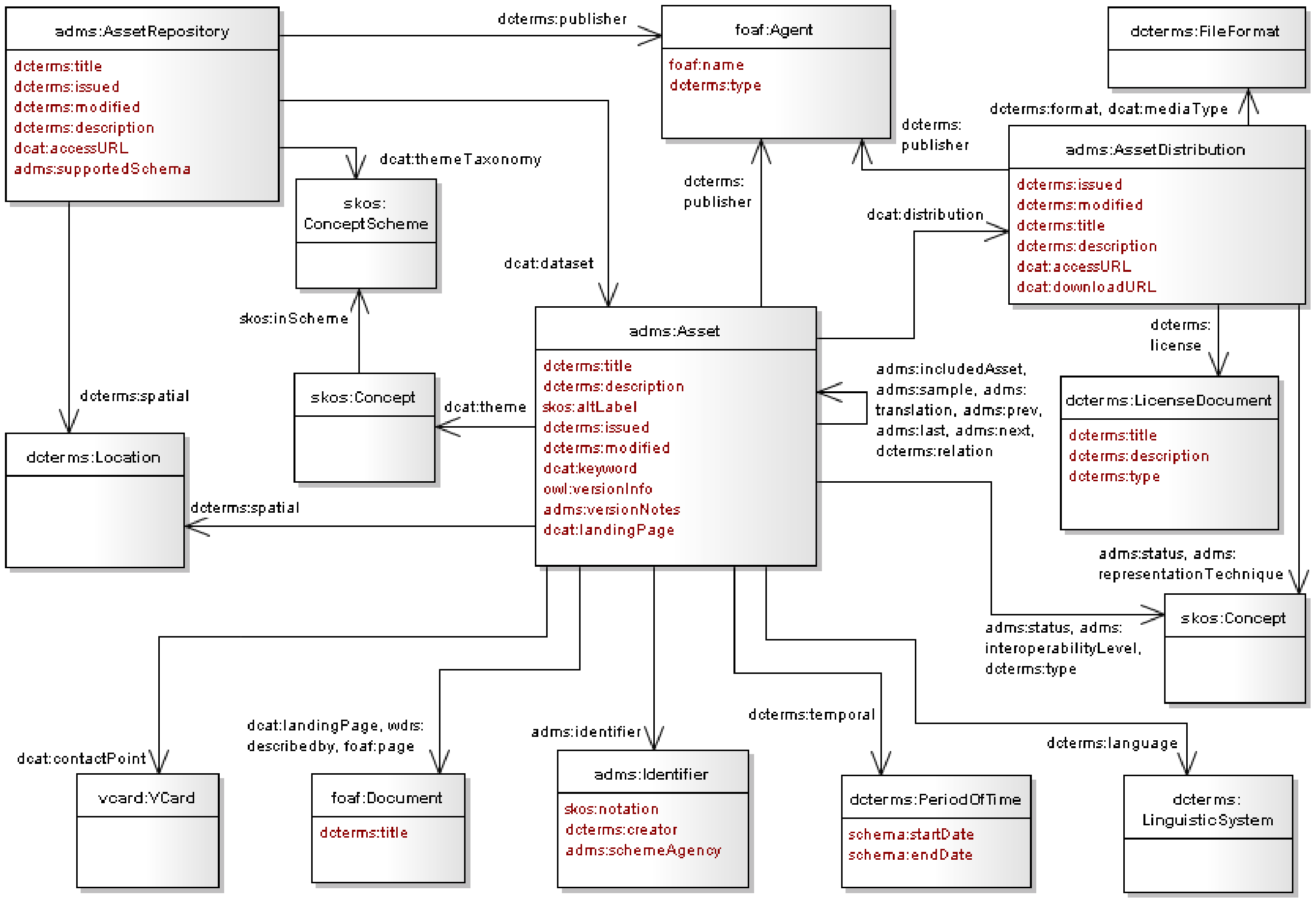
A spreadsheet template is provided, so a user can fill your dataset's metadata, and then Google Refine can create appropriate RDF.

The EU Open Data Portal uses a metadata vocabulary (EC-ODP) that's quite close to ADMS. For example, see the description of EuroVoc (EU's multilingual thesaurus), and the corresponding RDF file

3.5.3.1 W3C ADMS

After version 1.00, ADMS was contributed to W3C's Government Linked Data Working Group for further development. W3C streamlined the ADMS and converted it to a DCAT profile. This version of ADMS re-uses and subclasses DCAT and other standard vocabularies (e.g. SKOS, DCT) wherever possible and therefore defines a minimal set of classes and properties of its own. The ADMS specification was published on 1 August 2013 as a W3C note.

The domain model was simplified significantly. We work with this streamlined W3C version.

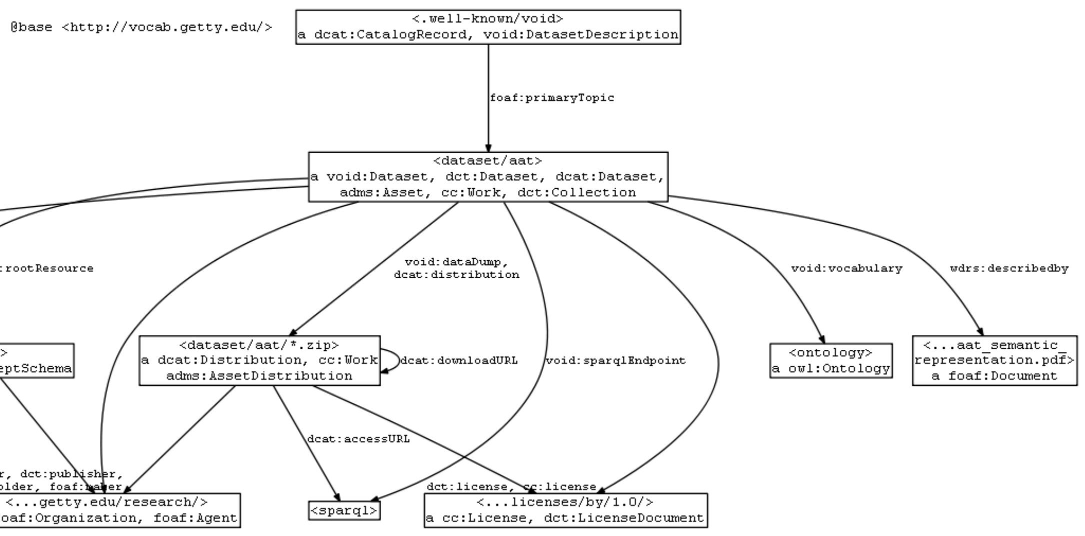


3.5.4 Descriptive Entities

We describe the following entities. Luckily, the domain models of VOID, DCAT and ADMS are fairly well aligned, so the entities can be assigned a consistent set of classes. Then the entities are interlinked with Descriptive Relations, more info is attached as Descriptive Properties, and more is computed as Dynamic Descriptive Properties.

entity	URL	classes
Descriptor	http://vocab.getty.edu/.well-known/void	void:DatasetDescription, dcat:CatalogRecord
AAT dataset	http://vocab.getty.edu/dataset/aat	void:Dataset, dct:Dataset, dcat:Dataset, adms:Asset, cc:Work, dct:Collection
Home page	http://vocab.getty.edu/	foaf:Document
AAT thesaurus	http://vocab.getty.edu/aat/	skos:ConceptSchema
Explicit Exports	http://vocab.getty.edu/dataset/aat/explicit.zip	dcat:Distribution, adms:AssetDistribution, cc:Work
Total Exports	http://vocab.getty.edu/dataset/aat/full.zip	
GVP ontology	http://vocab.getty.edu/ontology	owl:Ontology
Documentation	http://www.getty.edu/research/tools/vocabularies/lod/aat_semantic_representation.pdf	foaf:Document
Creator/publisher	http://www.getty.edu/research/	foaf:Organization, foaf:Agent
SPARQL endpoint	http://vocab.getty.edu/sparql	
License	http://opendatacommons.org/licenses/by/1.0/	cc:License, dct:LicenseDocument

A diagram of the entities and their relations follows:


@base <http://vocab.getty.edu/>
<.well-known/void> a dcat:CatalogRecord, void:DatasetDescription
foaf:primaryTopic
<dataset/aat> a void:Dataset, dct:Dataset, dcat:Dataset, adms:Asset, cc:Work, dct:Collection
dcat:landingPage, foaf:page, cc:attributionURL
void:rootResource
void:dataDump, dcat:distribution
void:vocabulary
wdrs:describedby
<> a foaf:Document
<aat> a skos:ConceptSchema
<dataset/aat/*.zip> a dcat:Distribution, cc:Work adms:AssetDistribution
dcat:downloadURL
void:sparqlEndpoint
<ontology> a owl:Ontology
<...aat_semantic_representation.pdf> a foaf:Document
dcat:accessURL
dct:creator, dct:publisher, dct:rightsHolder, foaf:maker
<...getty.edu/research/> a foaf:Organization, foaf:Agent
<sparql>
dct:license, cc:license
<...licenses/by/1.0/> a cc:License, dct:LicenseDocument

3.5.5 Descriptive Relations

The main descriptive entities are linked with the following relations:

subjects	relations	objects
http://vocab.getty.edu/.well-known/void	foaf:primaryTopic	http://vocab.getty.edu/dataset/aat
http://vocab.getty.edu/dataset/aat	dcat:landingPage, foaf:homepage, cc:attributionURL	http://vocab.getty.edu/
http://vocab.getty.edu/dataset/aat	wdrs:describedby	http://www.getty.edu/research/tools/vocabularies/lod/aat_semantic_representation.pdf
http://vocab.getty.edu/dataset/aat	void:vocabulary	http://vocab.getty.edu/ontology [2]
http://vocab.getty.edu/dataset/aat	void:rootResource	http://vocab.getty.edu/aat/ [3]
http://vocab.getty.edu/dataset/aat	void:sparqlEndpoint	http://vocab.getty.edu/sparql
http://vocab.getty.edu/dataset/aat	void:dataDump, dcat:distribution	http://vocab.getty.edu/dataset/aat/explicit.zip http://vocab.getty.edu/dataset/aat/full.zip
http://vocab.getty.edu/aat/ http://vocab.getty.edu/dataset/aat/explicit.zip http://vocab.getty.edu/dataset/aat/full.zip http://vocab.getty.edu/dataset/aat	dct:creator, dct:publisher, dct:rightsHolder, foaf:maker	http://www.getty.edu/research/
http://vocab.getty.edu/dataset/aat/explicit.zip	dcat:downloadURL	http://vocab.getty.edu/dataset/aat/explicit.zip
http://vocab.getty.edu/dataset/aat/full.zip		http://vocab.getty.edu/dataset/aat/full.zip
http://vocab.getty.edu/dataset/aat/explicit.zip http://vocab.getty.edu/dataset/aat/full.zip	dcat:accessURL	http://vocab.getty.edu/sparql

3.5.6 Descriptive Properties

We use the following descriptive properties and values.

- We use appropriate ADMS SKOS concepts from namespace http://purl.org/adms/ (Turtle). Their URLs are self-describing.
- The distribution of properties amongst entities is dictated by the domain models above (and see the actual VOID descriptor)

entity	property	values
descriptor	dct:title	"AAT description document (VOID file)"
descriptor	dct:created	"2014-03-20"^^xsd:date
descriptor	dc:format	"meta/void"
thesaurus	dct:title	"Art & Architecture Thesaurus (AAT) ®"
ontology	rdfs:label	"Getty Vocabulary Program ontology"
ontology	vann:preferredNamespacePrefix	"gvp"

[2] And the other External Ontologies used by GVP
[3] And all gvp:Facets, see next section

entity	property	values
ontology	vann:preferredNamespaceUri	"http://vocab.getty.edu/ontology#"
ontology	dc:format	"meta/rdf-schema"
publisher	foaf:name, rdfs:label	"Getty Research Institute"
dataset	dcat:contactPoint	[vcard:email <mailto:VocabLOD@getty.edu>]
publisher	dct:type	http://purl.org/adms/publishertype/NonProfitOrganisation
dataset	dct:title	"AAT Linked Open Data (LOD) Dataset"
dataset, ontology, exports	dct:created	"2014-02-20"^^xsd:date
dataset	dcat:keyword	"Thesauri"
dataset	dcat:theme, dct:subject	aat:300026677, http://id.loc.gov/authorities/subjects/sh85134827, http://dbpedia.org/resource/Thesaurus
dataset	void:exampleResource, adms:sample	aat:300264092, # Objects Facet aat:300264551, # Furnishings and Equipment (Hierarchy Name) aat:300197200, # <containers by function or context> aat:300198841. # Rhyta
dataset	dct:language	gvp_lang:en, gvp_lang:nl, gvp_lang:es, gvp_lang:zh [4]
dataset	voag:frequencyOfChange, dct:accrualPeriodicity	voag:BiWeekly [5]
dataset	dct:source	http://www.getty.edu/research/tools/vocabularies/aat/
dataset	dct:type	http://purl.org/adms/assettype/Thesaurus
dataset	adms:interoperabilityLevel	http://purl.org/adms/interoperabilitylevel/Semantic [6]
dataset	adms:representationTechnique	http://purl.org/adms/representationtechnique/SKOS
dataset	adms:status	http://purl.org/adms/status/Completed
dataset	vann:preferredNamespacePrefix	"aat"
dataset	vann:preferredNamespaceUri	"http://vocab.getty.edu/aat/"
dataset	void:uriSpace	"http://vocab.getty.edu/aat/"
dataset, ontology	owl:versionInfo	"1.0"
dataset	void:feature	fmt:N-Triples, fmt:RDF_XML, fmt:Turtle, fmt:SPARQL_Results_XML, fmt:SPARQL_Results_JSON
exports	dc:format	"application/n-triples", "application/zip"
	dcat:mediaType, dct:format	http://provenanceweb.org/format/mime/application/n-triples, http://provenanceweb.org/format/mime/application/zip (*)
explicit	dct:title	"Explicit AAT statements"

[4] Only the main AAT languages. Chinese (gvp_lang:zh) covers about 25% of all subjects)

[5] Frequency of data update and adding new Subjects. SDMX provides http://purl.org/linked-data/sdmx/2009/code#freq-W but that is "Weekly"

[6] See European Interoperability Framework (EIF)

entity	property	values
exports	dct:description	"NTriples zip, file size is approximate. First load ontologies, then files in indicated order"
	dcat:byteSize	"75000000"^^xsd:decimal
total exports	dct:title	"Total AAT statements"
	dct:description	"NTriples zip, file size is approximate"
	dcat:byteSize	"80000000"^^xsd:decimal

(*) Notes about dcat:mediaType, dct:format:

- Some VOID examples use types from http://mediatypes.appspot.com, e.g. http://purl.org/NET/mediatypes/application/n-triples
- For some reason "application/n-triples" is not in the IANA Media Types Registry, although it's given in the N-Triples spec. Consequently, neither http://provenanceweb.org/format/mime/application/n-triples nor http://purl.org/NET/mediatypes/application/n-triples are defined.
- We'd like to say void:feature fmt:N-Triples, but this property has domain void:Dataset, while the exports have type dcat:Distribution (object of void:dataDump).

3.5.7 License Info

Unless you are a lawyer, you may not care about licensing info. However, this info is important if you want to ensure you play by the rules, and some software uses it to filter to datasets satisfying certain open data criteria.

subject	property	object
http://vocab.getty.edu/dataset/aat http://vocab.getty.edu/dataset/aat/explicit.zip http://vocab.getty.edu/dataset/aat/full.zip	dct:license, cc:license	http://opendatacommons.org/licenses/by/1.0/
http://vocab.getty.edu/dataset/aat	dct:rights	"Copyright © 2000 The J. Paul Getty Trust. Made available under the ODC Attribution License"
	cc:attributionName	"Contains information from Art & Architecture Thesaurus (AAT)® which is made available under the ODC Attribution License"
	wv:norms	http://www.opendatacommons.org/norms/odc-by-sa/
	wv:declaration	"In circumstances where providing the full attribution statement is not technically feasible, the use of canonical AAT URIs is adequate to satisfy Section 4.3 of the ODC Attribution License"
http://opendatacommons.org/licenses/by/1.0/	dct:type	http://purl.org/adms/licencetype/Attribution
	cc:requires	cc:Attribution

3.5.8 VOID Subsets

As described in Per-Entity Exports, each independent entity (Subject, Source, Contributor) is available in several semantic formats. We express this as void:subsets with uriRegexPattern and the corresponding format:

```
<http://vocab.getty.edu/dataset/aat> void:subset
  <http://vocab.getty.edu/dataset/aat/subjects/rdf>.
<http://vocab.getty.edu/dataset/aat/subjects/rdf>
  dct:title "AAT Subjects as RDF/XML";
  void:uriRegexPattern "^http://vocab.getty.edu/aat/\\d+.rdf$";
  void:feature fmt:RDF_XML.
```

and similarly for:

- Sources and Contributors
- The other formats (fmt:N-Triples, fmt:Turtle, fmt:SPARQL_Results_JSON).

3.5.9 VOID Linksets

We also describe the AAT to LCSH Alignment following VOID section 5 Describing linksets.

```
<http://vocab.getty.edu/dataset/aat/alignment/lcsh> a void:Linkset;
  void:target
    <http://vocab.getty.edu/dataset/aat>,
    [a void:Dataset;
       dct:title "Library of Congress Subject Headings";
       foaf:homepage <http://id.loc.gov/authorities/subjects>];
  void:linkPredicate skos:exactMatch.
<http://vocab.getty.edu/dataset/aat> void:subset <http://vocab.getty.edu/dataset/aat/alignment/lcsh>.
```

- The alignment is a void:Linkset that connects two void:targets: AAT and LCSH using skos:exactMatch.
- We were not able to find a URL for the LCSH dataset: http://id.loc.gov/authorities/subjects provides a number of Alternate Formats, but these are "data dumps" or "distributions", not datasets
- So we describe it through its foaf:homepage. This is almost as good, as explained in VOID section 2.1 Web page links: "As foaf:homepage is an Inverse Functional Property, different descriptions of a dataset provided in different places on the Web can be automatically connected or “smushed” if they use the same homepage URI"
- The linkset is provided only as part of the AAT dataset and not separately (let us know if you want this changed). This is expressed by the last statement. Please note that the void:subset example in section 5.2 Linksets as part of larger datasets uses the wrong direction, see VOID Issue 105

We also count the void:triples in the linkset, see the end of next section.

3.5.10 Dynamic Descriptive Properties

Some of the properties need to be computed dynamically with every update (regeneration) of the dataset. We insert them in named graph http://vocab.getty.edu/.well-known/void, see next section:

- Set **dct:modified, dct:issued** of dataset, exports, descriptor to the dateTime of regeneration (now()). It's safe to assume that some data is updated on every generation; and publication will happen on the same day

```
insert {graph <http://vocab.getty.edu/.well-known/void> {
  <http://vocab.getty.edu/dataset/aat>              dct:modified ?date; dct:issued ?date.
  <http://vocab.getty.edu/dataset/aat/explicit.zip> dct:modified ?date; dct:issued ?date.
  <http://vocab.getty.edu/dataset/aat/full.zip>     dct:modified ?date; dct:issued ?date.
  <http://vocab.getty.edu/.well-known/void>         dct:modified ?date; dct:issued ?date.
}} where {bind (now() as ?date)}
```

- Declare all gvp:Facets as **void:rootResource** of the dataset, To let LOD crawlers find all statements in a top-down fashion. This is in lieu of declaring the Top Concepts of AAT

```
insert {graph <http://vocab.getty.edu/.well-known/void> {
  <http://vocab.getty.edu/dataset/aat> void:rootResource ?facet
}} where {?facet a gvp:Facet}
```

The rest provide VOID statistics (counts) about the dataset. More numbers than you can shake a stick at!

- Count total number of triples

```
insert {graph <http://vocab.getty.edu/.well-known/void> {
  <http://vocab.getty.edu/dataset/aat> void:triples ?count
}} where {select (count(*) as ?count) {?x ?p ?y}}
```

- Count number of main entities. We consider only gvp:Subject, not the subsidiary entities

```
insert {graph <http://vocab.getty.edu/.well-known/void> {
  <http://vocab.getty.edu/dataset/aat> void:entities ?count
}} where {select (count(*) as ?count) {?x a gvp:Subject}}
```

- Count number of distinct classes

```
insert {graph <http://vocab.getty.edu/.well-known/void> {
  <http://vocab.getty.edu/dataset/aat> void:classes ?count
}} where {select (count(*) as ?count)
   {select distinct ?class {?x a ?class. filter(!isBlank(?class))}}}
```

- Count number of distinct properties

```
insert {graph <http://vocab.getty.edu/.well-known/void> {
  <http://vocab.getty.edu/dataset/aat> void:properties ?count
}} where {select (count(*) as ?count) {select distinct ?prop {?x ?prop ?y}}}
```

- Count number of distinct subjects

```
insert {graph <http://vocab.getty.edu/.well-known/void> {
  <http://vocab.getty.edu/dataset/aat> void:distinctSubjects ?count
}} where {select (count(*) as ?count) {select distinct ?subj {?subj ?p ?y}}}
```

- Count number of distinct objects

```
insert {graph <http://vocab.getty.edu/.well-known/void> {
  <http://vocab.getty.edu/dataset/aat> void:distinctObjects ?count
}} where {select (count(*) as ?count) {select distinct ?obj {?x ?p ?obj}}}
```

- Count number of entities per class (class partition). In contrast to void:entities for the whole dataset, here we consider all entities.

```
insert {graph <http://vocab.getty.edu/.well-known/void> {
  <http://vocab.getty.edu/dataset/aat> void:classPartition
    [void:class ?class; void:entities ?count]
}} where {select ?class (count(*) as ?count)
    {?x a ?class. filter(!isBlank(?class))} group by ?class}
```

- Count number of triples per property (property partition)

```
insert {graph <http://vocab.getty.edu/.well-known/void> {
  <http://vocab.getty.edu/dataset/aat> void:propertyPartition
    [void:property ?prop; void:triples ?count]
}} where {select ?prop (count(*) as ?count) {?x ?prop ?y} group by ?prop}
```

- Count number of triples in Linkset (see previous section). We count only one direction ?x skos:exactMatch ?y but not the other direction ?y ?x nor the trivial statements ?x ?x and ?y ?y:

```
insert {graph <http://vocab.getty.edu/.well-known/void> {
  <http://vocab.getty.edu/dataset/aat/alignment/lcsh> void:triples ?count
}} where {select (count(*) as ?count) {?x skos:exactMatch ?y filter(str(?x) < str(?y))}}
```

3.5.11 VOID Deployment

The full descriptive info is available at the following URLs (VOID file in Turtle format):

1. http://vocab.getty.edu/.well-known/void, following the well-known URI specified in VOID spec section 7.2.
   - As you see above, this URL is used to tie the descriptor to the other entities
   - This URL is also linked from the GVP LOD home page
   - This URL is implemented as a HTTP 302 redirect to the following URL
2. http://vocab.getty.edu/void.ttl, following the practice in VOID spec section 6.2.

In addition, the descriptive info is available

3. In the repository, in its own named graph http://vocab.getty.edu/.well-known/void.

This follows the suggestion "query the endpoint itself in case it indexes its own VoID description" in section 2.1 of SPARQL Web-Querying Infrastructure: Ready for Action? (Carlos Buil-Aranda, Aidan Hogan, Jürgen Umbrich, and Pierre-Yves Vandenbussche, ISWC 2013) by the creators of the SPARQL Endpoint Status service. It is also queried by RKBExplorer VOID storage. The following key relations are used to discover the dataset and endpoint:

```
<http://vocab.getty.edu/.well-known/void>
  foaf:primaryTopic <http://vocab.getty.edu/dataset/aat>.
<http://vocab.getty.edu/dataset/aat>
  void:sparqlEndpoint <http://vocab.getty.edu/sparql>.
```

You can query the descriptive info using SPARQL, as shown at the end of section Counting and Descriptive Info

3.5.12 Possible Future Additions

We are considering the following additional descriptive info for future versions, and welcome your suggestions:

- Cover Vocabulary of a Friend (VOAF) metadata and register AAT at the Linked Open Vocabularies (LOV) site
- Provide SPARQL 1.1 Service Description of the endpoint, and tie up the VOID with it
- Provide VOID deployment through the endpoint URL, as per SPARQL 1.1 Service Description section 2
- void:openSearchDescription to describe the Full Text Search capabilities, following the Open Search 1.1 Specification
- We might consider VAEM, VDPP and VOAG, but we are not convinced they have significant penetration

3.6 Export Files

We provide export files (data dumps) in several configurations and formats.

3.6.1 Explicit Exports

These are statements in NTriples format, generated from GVP's database using R2RML.

- They are explicit statements only, so if you want to use them, you should also ensure the required Inference.
- First load the required External Ontologies (SKOS, SKOS-XL, ISO 25964): links are provided in that section
- Then load the GVP Ontology from http://vocab.getty.edu/ontology.rdf

Finally load the export files from http://vocab.getty.edu/dataset/aat/explicit.zip about 0.9Gb, 75Mb zipped)

```
AATOut_1Subjects.nt
AATOut_2Terms.nt
AATOut_AssociativeRels.nt
AATOut_ContribRels.nt
AATOut_Contribs.nt
AATOut_HierarchicalRels.nt
AATOut_Lang_sameAs.nt
AATOut_LCSHAlignment.nt
AATOut_Notations.nt
AATOut_ObsoleteSubjects.nt
AATOut_OrderedCollections.nt
AATOut_RevisionHistory.nt
AATOut_ScopeNotes.nt
AATOut_SemanticLinks.nt
AATOut_SourceRels.nt
AATOut_Sources.nt
```

- The files are named after the different parts of the semantic representation
- If loading to OWLIM, load Subjects first and Terms second (i.e. in the indicated order). OWLIM preserves the order of nodes as they are **first** inserted in the repository, and these files are sorted by the required Sort Order

The above is pretty much the actual process we use to load the AAT repository (SPARQL endpoint) with fresh data every 2 weeks.

3.6.2 Per-Entity Exports

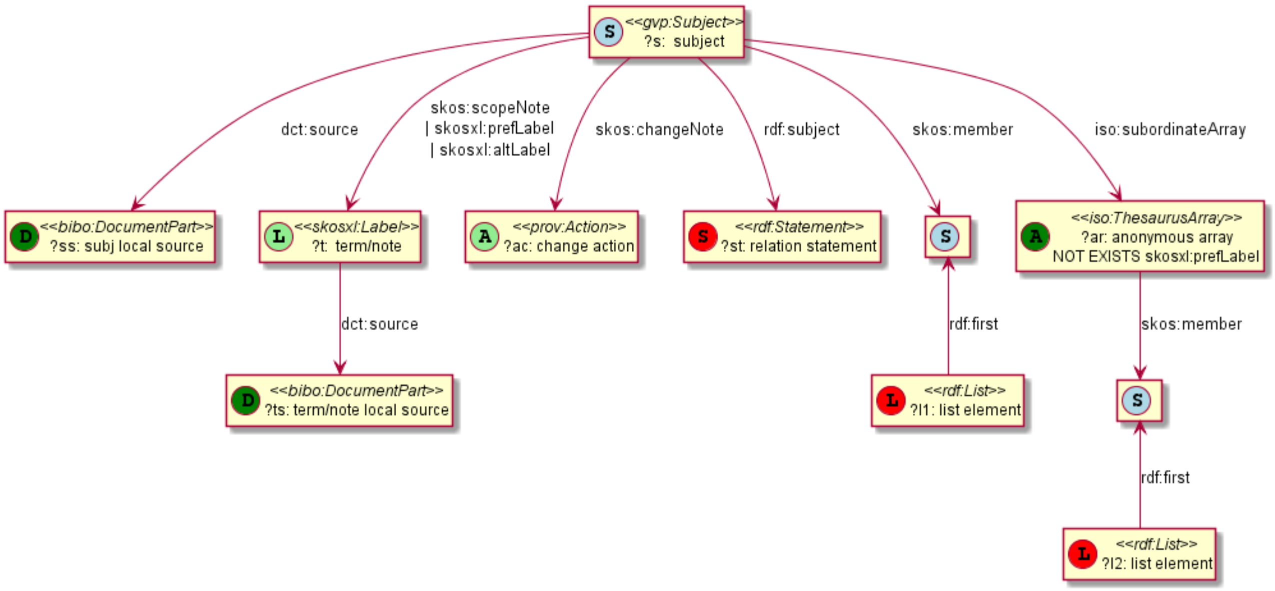
The Forest page for each resolvable URL provides downloadable semantic representations for that entity in RDF/XML, Turtle, NTriples, JSON formats. The same formats are available through content negotiation, and through direct URLs including file extension, as described in Semantic Resolution.

For the independent entities (Subjects, Sources, Contributors), the semantic formats include all triples (explicit and inferred) of all owned objects. This information is fetched with complex CONSTRUCT queries and cached for better performance.

There is no zip with all these files since they are over 80k. Use Total Exports instead, but if you want such a zip, please contact us.

The Construct Subject query is shown below and includes:

- All direct triples of the Subject (explicit and inferred) and the other nodes described below
- All local sources (bibo:DocumentPart)
- All terms and scope notes (skosxl:Label) and their local sources
- All skos:changeNote (prov:Activity)
- All rdf:Statements describing a relation where the subject plays the role of rdf:subject
- All rdf:List nodes of skos:member


S <<gvp:Subject>> ?s: subject
dct:source
skos:scopeNote | skosxl:prefLabel | skosxl:altLabel
skos:changeNote
rdf:subject
skos:member
iso:subordinateArray
D <<bibo:DocumentPart>> ?ss: subj local source
L <<skosxl:Label>> ?t: term/note
A <<prov:Action>> ?ac: change action
S <<rdf:Statement>> ?st: relation statement
S
A <<iso:ThesaurusArray>> ?ar: anonymous array NOT EXISTS skosxl:prefLabel
dct:source
rdf:first
skos:member
D <<bibo:DocumentPart>> ?ts: term/note local source
L <<rdf:List>> ?l1: list element
S
rdf:first
L <<rdf:List>> ?l2: list element

3.6.3 Total Exports

This file includes all statements (explicit and inferred) of all independent entities. It's a concatenation of the Per-Entity Exports in NTriples format. Because it includes all required Inference, you can load it to any repository (even one without RDFS reasoning). You still want to load the ontologies though, else you won't have descriptions of properties, associative relations, etc.

- First load the External Ontologies (SKOS, SKOS-XL, ISO 25964): links are provided in that section
- Then load the GVP Ontology from http://vocab.getty.edu/ontology.rdf
- Finally load the files from http://vocab.getty.edu/dataset/aat/full.zip (about 1.3Gb, 80Mb zipped)

```
AATOut_Full.nt     : subjects
AATOut_Contribs.nt : contributors
AATOut_Sources.nt  : sources
```

Please note that we have not tried out this process yet. If you encounter any problems, please contact us.

4 Sample Queries

In this section we provide sample queries for various tasks with the AAT LOD. We strive to have the same queries below the SPARQL search box in the Forest UI, but there might be some discrepancies.

If you write an interesting query, please contribute it!

4.1 Finding Subjects

4.1.1 Top-level Subjects

The top-level Subjects of AAT are gvp:Facets, so the query is easy:

```
select * {?f a gvp:Facet; skos:inScheme aat:}
```

4.1.2 Descendants of a Given Parent

300194567 "drinking vessels" as parent will bring back "rhyta" and other interesting records.

```
select * {?x gvp:broaderExtended aat:300194567}
```

4.1.3 Subjects by Contributor Abbrev

Subjects contributed by a particular contributor (e.g., GCI). You can find that contributor by foaf:nick

```
select * {
  ?x a gvp:Subject; dct:contributor ?contrib.
  ?contrib foaf:nick "GCI"}
```

4.1.4 Subjects by Contributor Id

If you know the Contributor id, you can use it directly. E.g. CGI is aat_contrib:10000088

```
select * {
  ?x a gvp:Subject; dct:contributor aat_contrib:10000088}
```

4.1.5 Preferred Ancestors

Fetch all preferred ancestors of 300226882 "baking dishes", together with their parents, very efficiently (no traversal). This can be used to reconstruct the hierarchy in memory.

```
select * {
  aat:300226882 gvp:broaderPreferredTransitive ?parent.
  OPTIONAL {?parent gvp:broaderPreferred ?grandParent}}
```

4.1.6 Full Text Search Query

This is the query used for the Full Text Search.

```
SELECT ?Subject ?Term ?Parents ?ScopeNote ?Type {
  ?Subject luc:term "fishing* AND vessel*"; a ?typ.
  ?typ rdfs:subClassOf gvp:Subject; rdfs:label ?Type.
  optional {?Subject gvp:prefLabelGVP [skosxl:literalForm ?Term]}
  optional {?Subject gvp:parentStringAbbrev ?Parents}
  optional {?Subject skos:scopeNote [dct:language gvp_lang:en; skosxl:literalForm ?ScopeNote]}}
```

- You should preprocess the query phrase for proper results, as **highlighted above** and described in section Full Text Search.
- For removing punctuation, it's important to use Unicode properties to decide what is not a letter/digit, lest you nuke Greek, Chinese hieroglyphs, and accented characters (e.g. Spanish, Dutch, Chinese transliterations).
- Use a powerful regexp handler and think \P{L} and \P{Nd}. Don't forget apostrophe.
- If you use this e.g. for an auto-completion application, be kind to our service and wait until the user has typed 3-4 chars and has waited a bit

4.1.7 Find Person Occupations

A user was searching for people occupations by luc:term (FTS) and further restriction s by

- FILTER regex(?parentStringAbbrev, "Agents Facet")
- FILTER langMatches(lang(?label), "en")

But of course there's a much faster way to restrict to a part of the GVP hierarchy: gvp:broaderExtended. If you've read this document carefully you know this, else explore the data but check the Inference dropdown!

1. Look at e.g. aat: 300025470 "presidents" with inference="Explicit and Implicit": http://vocab.getty.edu/aat/300025470?inference=all
2. Notice it has many gvp:broaderExtended:
   aat:300024978, aat:300024979, aat:300024980, aat:300025426, aat:300025427, aat:300025432, aat:300264089 (yay!)
3. How to pick the best root? You don't want to explore all them broaders one by one.
4. Click on the Hierarchy tab, which sends you to the Getty site.
   (The semantic site does not have a decent hierarchical display yet)
5. aat:300024979 "people (agents)" looks ok. But aat:300024980 <people by occupation> is exactly what we need!
6. Use gvp:prefLabelGVP (instead of lang "en") because some concepts may NOT have an "en" prefLabel
7. Don't forget a wildcard in luc:term, or you'll miss "presidents' wives"
8. Consider NOT ordering by ?label, since luc:term natively returns them in relevance order:

• Order by ?label:	first ladies, presidents, vice-presidents
• By Lucene relevance:	presidents, vice-presidents, first ladies

The rewritten query is short, nice and efficient:

```
select * {
?c a gvp:Concept;
   gvp:broaderExtended aat:300024980 ; # <Persons by Occupation>
   gvp:prefLabelGVP/xl:literalForm ?label ;
   luc:term "president*"}
```

You may also try with "engraver*"

4.1.8 Find Subject by Exact English PrefLabel

The FTS query above is perfect for auto-completion services, i.e. for returning many potential matches when the user enters a keyword (perhaps partial). But if you are working on co-referencing (e.g. with OpenRefine reconciliation), you may need something simpler: match by exact label:

```
select * {?subj gvp:prefLabelGVP/xl:literalForm "rhyta"@en}
```

Note that prefLabelGVP is usually in the plural!

It's important to specify the language tag, else the query will return nothing. Most prefLabelGVP are in English (but not all, see next section)

4.1.9 Find Subject by Language-Independent PrefLabels

Most prefLabelGVP are in English but not all. 1329 are in a different language:

```
select * {?subj gvp:prefLabelGVP/xl:literalForm ?label. filter(lang(?label) != "en")}
```

Of these exceptions, almost all are in en-us "American English" (curiously, even "IEEE 802.11"@en-us is marked so).

And there's one in Spanish: aat:300181273 "troffers"@es (a kind of recessed luminaires/lamps).

The point is, you may prefer to search by string without providing the lang tag:

```
select distinct ?subj {?subj skos:prefLabel ?label. filter(str(?label)="rhyta")}
```

- We use str() to strip off the lang tag
- We search for all prefLabels (not just prefLabelGVP)
- We use the shortcut skos:prefLabel instead of having to go through xl:literalForm
- We have to add "distinct" because a subject may have the same string in two languages, e.g. "rhyta"@en and "rhyta"@el-latn (the latter is Greek transliterated to Latin)

4.1.10 Find Subject by Any Label

Since GVP prefers the Plural form for Descriptor term (prefLabel), the previous query finds nothing for "rhyton".

To widen the search to cover all labels, we add skos:altLabel to the query. We use "|". which is a SPARQL 1.1 Property Path feature.

```
select distinct ?subj {?subj skos:prefLabel|skos:altLabel ?label. filter(str(?label)="rhyton")}
```

4.1.11 Find Terms by Language Tag

Say you want to find all Berber terms transliterated to Latin (no? I do this every day!):

```
select * {
  ?subj xl:prefLabel|xl:altLabel [xl:literalForm ?term]
  filter(langMatches(lang(?term),"ber-Latn"))}
```

- Here we use query features we've seen before: a property path alternative "|", and a blank node since we don't care about the xl:Label but only about its literal form.
- We also use a new feature: langMatches(). It lets us find not only terms in Berber (e.g. "Zemmour"@ber-latn), but also in Berber Dialects (e.g. "Ait Youssi"@ber-latn-x-dialect).

You could try to use substr(lang()) or strstarts(lang()), but the comparison must be case-insensitive (see Language Tag Case). langMatches() is easier to use and is dedicated to this purpose.

4.1.12 Find Ordered Subjects

Find subjects whose children have "forced" (explicit) ordering.

- All subjects have gvp:displayOrder, so we check for non-trivial one (>1)
- We get the label using an anonymous node notation "[..]"

```
select ?coll ?label {
  ?coll gvp:prefLabelGVP [skosxl:literalForm ?label]
  filter(exists {?coll gvp:narrower [gvp:displayOrder ?order] filter(?order>1)})}
```

4.1.13 Find Ordered Hierarchies

In this section we do something similar as the previous one, but illustrate some different approaches:

- Find ordered hierarchies (gvp:Hierarchy) by explicit type (skos:OrderedCollection). The semantic representation uses skos:OrderedCollection for exactly those subjects whose children have non-trivial "forced" ordering
- We get the label using SPARQL Property Path syntax "/"
- We use the shorter prefix "xl:" instead of "skosxl:": both of these prefixes are defined in the repository, and mean the same

```
select * {?x a gvp:Hierarchy, skos:OrderedCollection; gvp:prefLabelGVP/xl:literalForm ?label}
```

4.1.14 Get Subjects in Order

Some subjects e.g. 300020605 "Pompeian wall painting styles" have children laid out in a particular order:

```
select * {
  aat:300020605 gvp:narrower ?x.
  optional {?x gvp:displayOrder ?s}.
  ?x gvp:prefLabel [xl:literalForm ?l]
} order by ?s
```

Because OWLIM preserves the order in which resources are first encountered, subjects and terms are returned in the desired order, even if you don't use the custom field gvp:displayOrder in your query:

```
select * {
  aat:300020605 gvp:narrower ?x.
  ?x gvp:prefLabel [xl:literalForm ?l]}
```

4.2 Getting Information

4.2.1 Subject Preferred Label

For each of the Find queries, you can get the preferred label in addition to the URI by adding this fragment:

```
 ?x gvp:prefLabelGVP [skosxl:literalForm ?label]
```

- Here we reach to the skosxl:Label preferred by GVP (each Subject has exactly one). Since we don't care about the skosxl:Label node (only about the label stored there), we use a blank node in the query.

Eg when we add this fragment to "Subjects by Contributor Abbrev", we get this query:

```
select * {
  ?x a gvp:Subject; dct:contributor ?contrib;
     gvp:prefLabelGVP [skosxl:literalForm ?label].
  ?contrib foaf:nick "GCI"}
```

4.2.2 Preferred and Vernacular

Find concepts having alternate labels in the Vernacular, and list them together with the preferred label.
(Please note that there are some preferred labels that are also in the Vernacular.)

```
select ?c ?preferred ?vernacular {
  ?c xl:altLabel ?t; gvp:prefLabelGVP/xl:literalForm ?preferred.
  ?t gvp:termFlag <http://vocab.getty.edu/term/flag/Vernacular>; xl:literalForm ?vernacular}
```

4.2.3 Scientific Names by Language

"Scientific name" is the zoology/botany taxon name (genus, species, etc) and comes from Latin. However, all such terms are also emitted in English. I wondered what is the distribution of scientific names across languages:

```
select ?lang (count(*) as ?c) {
  ?t gvp:termKind <http://vocab.getty.edu/aat/term/kind/ScientificOrTechnical>; dct:language ?l.
  ?l gvp:prefLabelGVP/xl:literalForm ?lang
} group by ?lang
```

The result for AAT is:

lang	**c**
English (language)@en	1403
Latin (language)@en	1351
Spanish (language)@en	63
Italian (language)@en	1
French (language)@en	1

4.2.4 Scientific Names not in English and Latin

Then I wondered which are the odd scientific names (neither English nor Latin):

```
select ?c ?lab {
  ?c xl:prefLabel|xl:altLabel ?t.
  ?t gvp:termKind <http://vocab.getty.edu/term/kind/ScientificOrTechnical>;
     xl:literalForm ?lab.
  filter (lang(?lab) not in ("en", "la"))}
```

4.2.5 Construct Subject

This complex query is used to get all info about a Subject and its owned sub-objects (see Per-Entity Exports)

```
CONSTRUCT {
  ?s  ?p1 ?o1. # subject
  ?ac ?p2 ?o2. # change action
  ?t  ?p3 ?o3. # term/note
  ?ss ?p4 ?o4. # subject local source
  ?ts ?p6 ?o6. # term/note local source
  ?st ?p7 ?o7. # statement about relations of subject
  ?ar ?p8 ?o8. # anonymous array of subject
  ?l1 ?p9 ?o9. # list element of subject
  ?l2 ?p0 ?o0. # list element of anonymous array
} WHERE {
  BIND (aat:300107346 as ?s)
  {?s ?p1 ?o1 FILTER(!isBlank(?o1))}
  UNION {?s skos:changeNote ?ac. ?ac ?p2 ?o2}
  UNION {?s dct:source ?ss. ?ss a bibo:DocumentPart. ?ss ?p4 ?o4}
  UNION {?s skos:scopeNote|skosxl:prefLabel|skosxl:altLabel ?t.
     {?t ?p3 ?o3 FILTER(!isBlank(?o3))}
     UNION {?t dct:source ?ts. ?ts a bibo:DocumentPart. ?ts ?p6 ?o6}}
  UNION {?st rdf:subject ?s. ?st ?p7 ?o7}
  UNION {?s skos:member/^rdf:first ?l1. ?l1 ?p9 ?o9}
  UNION {?s iso:subordinateArray ?ar FILTER NOT EXISTS {?ar skosxl:prefLabel ?t1}.
     {?ar ?p8 ?o8}
     UNION {?ar skos:member/^rdf:first ?l2. ?l2 ?p0 ?o0}}
}
```

Blank nodes (owl:Restriction types) are possible for Subject, Term and Note (?o1 and ?o3), so we FILTER them out.

4.2.6 Historic Information on Relations

Here is an example query to fetch all relations of aat:300020271, together with optional Historic Information.

```
SELECT ?concept1 ?rel ?concept2 ?startDate ?endDate ?comment ?hist {
  ?concept1 ?rel ?concept2. FILTER(?concept1=aat:300020271 || ?concept2=aat:300020271)
  ?concept1 a gvp:Subject. ?concept2 a gvp:Subject.
 OPTIONAL {
    ?statement a rdf:Statement;
      rdf:subject ?concept1;
      rdf:predicate ?rel;
      rdf:object ?concept2.
    OPTIONAL {?statement schema:startDate ?startDate}.
    OPTIONAL {?statement schema:endDate ?endDate}.
    OPTIONAL {?statement rdfs:comment ?comment}.
    OPTIONAL {?statement gvp:historicFlag ?hist}}}
```

We show some of the results below (extensive SKOS Inference is involved, so we omit a lot of the inferences):

concept1	Rel	concept2	startDate	endDate	comment	hist
aat:300020271	gvp:aat2812_followed	aat:300020269	-2785	-2765	Second Dynasty began ca. 2775 BCE	-
aat:300020269	gvp:aat2811_preceded	aat:300020271	-2785	-2765	Second Dynasty began ca. 2775 BCE	-
aat:300020271	gvp:broaderPreferred	aat:300020265	-	-	-	-
aat:300020271	skos:broader	aat:300020265	-	-	-	-

4.2.7 Historic Information of Terms

Fetch all terms with Historic Information, together with that information:

```
select ?term ?literal ?start ?end ?comment {
  ?term a xl:Label; xl:literalForm ?literal; dct:valid ?x.
  optional {?term schema:startDate ?start}
  optional {?term schema:endDate ?end}
  optional {?term rdfs:comment ?comment}}
```

4.2.8 Preferred Terms for Contributors

Terms that are preferred for specific contributors. We want only the labels, so we use blank nodes […] to disregard the URIs. You can similarly use gvp:contributorNonPreferred and gvp:contributorAlternatePreferred.

```
select * {[xl:literalForm ?term] gvp:contributorPreferred [foaf:nick ?contrib]}
```

4.2.9 Preferred Terms for Sources

Terms that are preferred for specific sources. We want only the labels, so we use blank nodes […] to disregard the URIs. You can similarly use gvp:sourceNonPreferred and gvp:sourceAlternatePreferred.

```
select * {[xl:literalForm ?term] gvp:sourcePreferred [bibo:shortTitle ?source]}
```

4.2.10 Concepts Related by Particular Associative Relation

While investigating the precise meaning of the Associative Relationship 2100 "distinguished from", we tried this query. It retrieves related concepts, their labels and scope notes (in English).

- We use blank nodes "[…]" instead of property paths "/" because of a SPARQL parser bug (SES-2024)
- We filter by the string representation of the two concept URIs because this relation is symmetric, and we don't want to get it both forward and inverse in the result set.

```
select * {
  ?c1 gvp:prefLabelGVP [xl:literalForm ?l1];
    skos:scopeNote [xl:literalForm ?n1; dct:language gvp_lang:en].
  ?c1 gvp:aat2100_distinguished_from ?c2. filter (str(?c1) < str(?c2))
  ?c2 gvp:prefLabelGVP [xl:literalForm ?l2];
    skos:scopeNote [xl:literalForm ?n2; dct:language gvp_lang:en]}
```

4.2.11 Languages and ISO Codes

We use the Guide Term 300389738 <languages and writing systems by specific example> and the two specific sources described at Language Tags and Sources to return all languages together with their ISO2 and ISO3 language codes (where assigned):

```
select ?lang ?name ?iso2 ?iso3 {
  ?lang gvp:broader aat:300389738; gvp:prefLabelGVP/skosxl:literalForm ?name.
  optional {?lang skosxl:altLabel [skosxl:literalForm ?iso2; dct:source aat_source:2000075479]}
  optional {?lang skosxl:altLabel [skosxl:literalForm ?iso3; dct:source aat_source:2000075493]}}
```

Some observations on the ISO codes:

- Only the "more important" languages have alpha-2 codes (e.g. Abkhaz has one, but Abaza doesn't)
- Out of 1883 languages, 1330 (70%) have an alpha-3 code. The rest are more exotic languages, variants (e.g. transliterated; liturgical; medieval), or modifications (e.g. GVP uses Frisian. ISO has defined codes for its predecessor Old Frisian and its dialects West, Saterland and North Frisian, but not for Frisian itself).

4.2.12 Language URLs

Find all "logical" language URLs (Language Dual URLs):

```
select * {?x owl:sameAs ?y FILTER(str(?x) > str(?y))} order by str(?x)
```

You **must** check "Expand results over equivalent URIs" in the SPARQL UI. Or use this direct query link.

4.3 Counting and Descriptive Info

Counts can give you a better feel of the available semantic information.

4.3.1 Descriptive Info from VOID

As an alternative to getting the VOID descriptive info as a Turtle file (see VOID Deployment), you can query it in the repository. The following query returns all descriptive triples (same as the Turtle file):

```
select * {graph <http://vocab.getty.edu/.well-known/void> {?s ?p ?o}}
```

4.3.2 Number of Entities from VOID

Get the class partitions (number of entities per class), ordering by decreasing count:

```
select ?class ?count {?void void:classPartition [void:class ?class; void:entities ?count]}
order by desc(?count)
```

This uses the pre-calculated counts in the VOID descriptive info, so it's very fast.

4.3.3 Number of Entities (Dynamic)

Some of the class partitions (see previous section) don't return very meaningful numbers:

- xl:Label returns both terms and scope notes, but you probably want them separately
- bibo:Document returns both local and global sources

We can get those numbers with the queries below, though this is slower:

- Terms

```
select (count(*) as ?c) {?x skosxl:prefLabel|skosxl:altLabel ?t}
```

- Scope Notes

```
select (count(*) as ?c) {?x skos:scopeNote ?t}
```

- Local Sources

```
select (count(*) as ?c) {?x a bibo:DocumentPart}
```

- Global Sources

```
select (count(*) as ?c) {
  ?x a bibo:Document
  filter not exists {?x a bibo:DocumentPart}}
```

4.3.4 Number of Revision Actions

Count of Revision History actions (Create, Modify, Publish) by ?action kind and entity ?type

```
select ?action ?type (count(*) as ?c) {
  ?x skos:changeNote ?y. ?y dc:type ?action
  bind(if(exists{?x a bibo:Document},"Source","Subject") as ?type)
} group by ?action ?type
```

4.4 Explore the Ontology

The GVP Ontology is described in this document, and a reference is provided in the ontology documentation (namespace document). But you can also explore it with queries.

4.4.1 Ontology Classes and Properties

This returns the classes and properties defined by the ontology, together with some details:

```
select ?x ?type (coalesce(?descr,?label) as ?description) ?domain ?range {
  ?x rdfs:isDefinedBy <http://vocab.getty.edu/ontology>; a ?type.
  optional {?x dct:description ?descr}
  optional {?x rdfs:label ?label}
  optional {?x rdfs:domain ?domain}
  optional {?x rdfs:range ?range}}
```

- Uncheck "Include inferred" so you don't get duplicate rows due to super-types.

- coalesce() returns the more detailed dct:description if available, else rdfs:label

4.4.2 Ontology Values

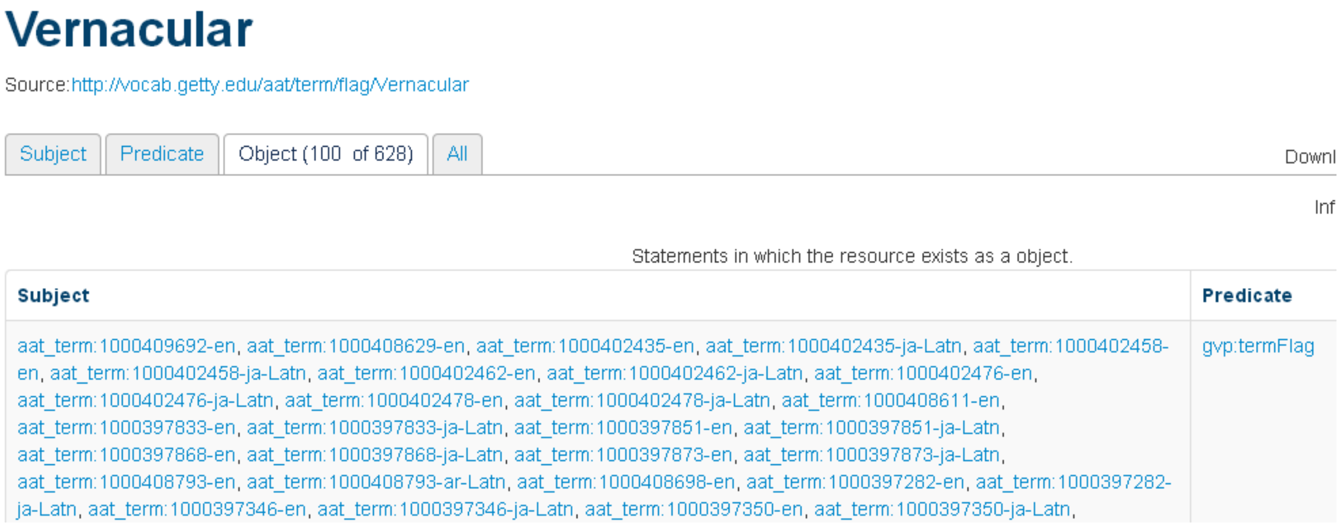
This returns all values (skos:Concepts, mostly Term Characteristics) in small schemes defined by the ontology:

```
select * {
  ?x skos:inScheme [rdfs:isDefinedBy <http://vocab.getty.edu/ontology>; rdfs:label ?scheme];
     skos:prefLabel ?value; skos:scopeNote ?note; skos:example ?example}
```

Because the scheme URL is always a prefix of the value URL, we print the scheme's label instead. Some example results (the prefix http://vocab.getty.edu/ is omitted):

x	**scheme**	**value**	**note**	**example**
term/flag/Vernacular	Term Flag	Vernacular	Term is in the "vernacular" language	"Firenze" is the vernacular in Italian (TGN)
term/kind/Abbreviation	Term Kind	Abbreviation	Term is an abbreviation, initialism, or acronym	DVD, CD-ROM (AAT)
term/kind/CommonTerm	Term Kind	Common term	Preferred common language term. Used for subjects that also include a Scientific term	domestic cat (AAT)
term/kind/ScientificOrTechnical	Term Kind	Scientific or Technical term	A Scientific term	"Felis domesticus" is the scientific term for "cats" (AAT)

You can click on the value URLs (first column) and then the Object tab to explore terms having that characteristic. For example, there are 628 AAT terms that have Term Flag "Vernacular":



Statements in which the resource exists as a object.

Subject	Predicate
aat_term:1000409692-en, aat_term:1000408629-en, aat_term:1000402435-en, aat_term:1000402435-ja-Latn, aat_term:1000402458-en, aat_term:1000402458-ja-Latn, aat_term:1000402462-en, aat_term:1000402462-ja-Latn, aat_term:1000402476-en, aat_term:1000402476-ja-Latn, aat_term:1000402478-en, aat_term:1000402478-ja-Latn, aat_term:1000408611-en, aat_term:1000397833-en, aat_term:1000397833-ja-Latn, aat_term:1000397851-en, aat_term:1000397851-ja-Latn, aat_term:1000397868-en, aat_term:1000397868-ja-Latn, aat_term:1000397873-en, aat_term:1000397873-ja-Latn, aat_term:1000408793-en, aat_term:1000408793-ar-Latn, aat_term:1000408698-en, aat_term:1000397282-en, aat_term:1000397282-ja-Latn, aat_term:1000397346-en, aat_term:1000397346-ja-Latn, aat_term:1000397350-en, aat_term:1000397350-ja-Latn,	gvp:termFlag