Table of Contents

Experience xAPI Vocabulary Primer

(send questions or feedback to: xapi-vocabulary@adlnet.gov)

> An accessible, printer-ready and reader-friendly version of this companion document is also available as a gitbook here: https://www.gitbook.com/book/adl/experience-xapi-vocabulary-primer/details

Status of this Document

Owner

Name	Website	Email
xAPI Vocabulary Working Group	https://www.w3.org/community/xapivocabulary	xapi-vocabulary@adlnet.gov

Revision history

Version	Issue date	Author/editor	Description/Summary of changes
1.0	8/07/2015	Russell Duhon	Initial Draft/Original Quip document written by Russell Duhon
1.0.1	10/19/2015	Jason Haag	Second Draft/Updated information and examples to reflect existing vocabularies published by the community
1.0.2	1/5/2016	Jason Haag	Major edits to consolidate content and align with verbiage of Companion Spec
1.0.2	2/5/2016	Russell Duhon	Edits to clean up examples

Reviewed by

Version	Review Date	Name	Organization	Notes
1.0.2	2/3/2016	Tom Creighton	ADL	
1.0.2	2/4/2016	Leslie Spotswood	ADL	
1.0.2	2/5/2016	Russell Duhon	Saltbox	

Related documents

Document	Location
Experience API (xAPI) Companion Specification for Vocabularies	Github

License

How to Read this Document

This Primer is an introductory document designed to complement the Companion Specification for xAPI Vocabularies by providing the reader with additional knowledge and examples. It is more precise about how to use the terms than the Companion Specification. It also covers some introductory real-world examples of what can be immediately achieved by publishing xAPI vocabularies as Linked Data (LD). The reader is presented with a descriptive narrative of classes and properties that are most commonly used for representing xAPI vocabularies as linked datasets.

Formatting

Properties and classes are **bolded**.

Code examples are in `red text or have a light gray background`.

Vocabularies and vocabulary term identifiers are all ***`red bold italicized`*** when described in this document's text.

Internal hyperlinks are blue text and external links are ***bold blue italicized***.

Several examples will be also provided with code snippets contained inside of a gray box like the one below. Most of the examples in this document are initially written using the ***Turtle syntax for RDF***. Turtle is used here for improved readability as it allows RDF to be completely written in a compact and simplified natural text form. Examples serialized as Turtle and JSON-LD will appear throughout this document as sample code contained inside the gray box such as the following:

```
#Turtle example
@prefix ex: <http://www.example.com/>.

ex:aResource ex:aProperty ex:anotherResource;
  ex:anotherProperty "An RDF Literal"@en.
```

Each resource is described as RDF triples as discussed in the Companion Specification for xAPI Vocabularies. A simplified example of using Turtle for the xAPI Verb 'satisfied' with a basic label and description would be written as:

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix skos: <http://www.w3.org/2004/02/skos/core#> .
@prefix xapi: <https://w3id.org/xapi/ontology#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

<https://w3id.org/xapi/adl/verbs/satisfied> a xapi:Verb ;
    skos:prefLabel "satisfied"@en ;
    skos:definition "Indicates that the Authority or application determined the Actor
has fulfilled the criteria of the Activity."@en .
```

Some examples in this primer will also be provided as JSON-LD. The above Turtle serialization is equivalent to the following example, in JSON-LD syntax:

```
{
  "@context": {
    "rdf": "http://www.w3.org/1999/02/22-rdf-syntax-ns#",
    "rdfs": "http://www.w3.org/2000/01/rdf-schema#",
    "skos": "http://www.w3.org/2004/02/skos/core#",
    "xapi": "https://w3id.org/xapi/ontology#",
    "xsd": "http://www.w3.org/2001/XMLSchema#"
  },
  "@id": "https://w3id.org/xapi/adl/verbs/satisfied",
  "@type": "xapi:Verb",
  "skos:prefLabel": {
    "@language": "en",
    "@value": "satisfied"
  },
  "skos:definition": {
    "@language": "en",
    "@value": "Indicates that the Authority or application determined the Actor has fu
lfilled the criteria of the Activity."
  }
}
```

Introduction and Motivations

In the core ***Experience API (xAPI) Specification***, there are a lot of things identified by using an ***Internationalized Resource Identifier*** (IRI). When the core xAPI specification was written it was envisioned that the IRIs used for identifiers in xAPI statements could eventually map directly to vocabulary IRIs for more semantic meaning. Most IRIs will be URLs, just like you commonly see in web browsers. Using IRIs for identifiers gives many advantages, such as clear rules that prevent people from trampling on each other's identifiers, having a place to go to find out more built in, and interoperability with other standards that use IRIs.

One of those standards is called the ***Resource Description Framework*** (RDF), and it's part of the Semantic Web. A basic introduction to RDF is provided in the Companion Specification for xAPI Vocabularies. The Semantic Web is a loose collection of standards and practices, including for querying stores of data, for exchanging data in a multitude of formats, for creating ontologies — descriptions of how patterns relate to other patterns, so that you can infer new meanings. All of these capabilities are built on top of or meant to work heavily with RDF.

This means it is very useful to have ways to talk about parts of xAPI that are identified by IRIs in RDF. By doing so, many powerful tools for working with RDF, such as specialized databases, automatic inference engines, and code libraries in virtually every programming language that have built up over many years can be brought to bear. What's more, many of the things we might want to say in the xAPI community, such as that one verb is a more specific form of another, can be expressed using existing schemas and ontologies based on RDF.

For example, some of the types of questions that will be not just possible, but rather easy for us to answer:

1. Has anyone published a translation of any existing xAPI verbs in Simplified Chinese (or any other language)?
2. How many published xAPI vocabularies are reusing a particular Verb or Activity Type?
3. What verbs are more specialized forms of a verb, in any vocabulary?

The main downside of RDF has historically been that working with it in programs requires importing quite a bit of complexity. Luckily, another standard provides a highly effective bridge between the higher complexity world of RDF and the simplicity of JSON (the same format xAPI Statements are written in): JSON-Linked Data (JSON-LD). People wanting to

generate descriptions of xAPI verbs, activity types, and so forth can initially work with simple HTML/RDFa representations, and anyone wanting to deliver additional value can transform HTML/RDFa representations into JSON-LD.

Whirlwind Backdrop

Instead of defining a whole new batch of properties and classes, the more common approach is to reuse existing, proven ontologies and schemas where appropriate, adopting a minimal set of new properties and classes where they provide useful structure. That’s the widely accepted way of using RDF in situations of this sort.

The next sections will cover the following:

- The classes and properties that can be used for describing xAPI vocabularies.
- When to use them and some use cases and scenarios they enable.
- Examples of a vocabulary shown as Turtle RDF and JSON-LD.
- How to use SPARQL queries showing a bit of what becomes possible.

So far, almost all the work on using RDF with xAPI has focused on Verbs. As a result, Verbs will be the main focus of the examples in this document, but it could expand with time to include extensions, activities themselves, agents, and attachment usage types. As that happens, there will probably be a document for each of these scenarios.

Vocabulary Classes

Most of the classes used for describing xAPI vocabularies come from the Simple Knowledge Organization System (SKOS). Read the Companion Specification for xAPI Vocabularies for more details on the intentional alignment with SKOS. This section will briefly describe all of the classes used for xAPI vocabularies.

skos:ConceptScheme is the class for collections, lists, and other similar types of a controlled vocabulary. In the xAPI sense, it will generally be used for various collections of verbs or activity types, particularly represented as vocabularies.

xapi:ActivityType is the class used for xAPI Activity Types. It is a ‘subclass’ of **skos:Concept**.

xapi:Verb is the class for xAPI Verbs. It is a ‘subclass’ of **skos:Concept**.

prov:Activity is a class used in connection with vocabulary provenance properties.

Vocabulary Properties

Now, on to properties. These will generally require more explanation as to their appropriate use. The classes above will mostly come up in relation to using these properties. Properties that hold natural language should always use language-tagged strings, so that systems looking to display them to speakers of various languages will be able to work with them.

Properties used for labeling

skos:prefLabel should only be used by the creator of a vocabulary term (Verb or Activity Type). It holds the single (per language tag) preferred form for the vocabulary term's display. If you're coining a new Verb or Activity Type and want to suggest a display for it, use **skos:prefLabel**.

skos:altLabel should only be used (directly) by the creator of a vocabulary term. It holds alternate forms for the vocabulary term's displayed label. This won't come up too often unless there are two or more strong candidates for what to use in a vocabulary term's display.

xapi:thirdPartyLabel is a "sub property" of **skos:altLabel**. It is intended to be used by people who did not create the original vocabulary term, but would like to offer an alternative label for it. This could also be used in a language label that already exists, but doesn't 'read well' with some data, or in a language where there isn't yet a label. For example, a system that was displaying a Verb to a Chinese speaker might first check for a **skos:prefLabel** in Traditional Chinese, then an **skos:altLabel**, then a **xapi:thirdPartyLabel**. This property would also be useful when there's a Verb with the proper meaning, but without a suitable display.

Properties for adding descriptions, notes, and related information directly

skos:definition is used to define and describe a Verb (or other xAPI thing) in natural language. It should be left to the creator, along with most of the documentation properties below, though there may be occasions, such as translation, where it makes sense for someone who didn't create the verb to provide a definition.

skos:example connects an xAPI Verb or Activity Type to an example of its use, most likely as a xAPI Statement example. To do that, use a blank node with a **prov:value** holding a JSON representation of the Statement. People other than the creator should feel free to provide examples for use of verbs in new situations.

skos:historyNote, skos:scopeNote, & **skos:editorialNote** all have their uses, but are fairly specialized. Some things they cover include: historical information on change of a term's nature, usage notes regarding scope of application, and information for vocabulary maintainers, respectively. In the unlikely event none of those fit, **skos:note** is your last resort. Most of these should be left to the creator, generally. One reasonable non-creator use would be adding a short **skos:scopeNote** to a referenced verb, giving a brief description of its use in relation to the referencing vocabulary. The **skos:scopeNote** should always include a description of the specialized purpose (e.g. "The IEEE ADB vocabulary uses ADL's 'answered' verb for providing the answer to an interactive question within the eBook or an assessment."), as there may be quite a few of them for the most often referenced verbs.

Properties for connecting terms to the vocabulary

skos:inScheme connects an **xapi:Verb** or other xAPI term maintained as part of a vocabulary (**xapi:ActivityType**) to the **skos:ConceptScheme** that is the xAPI vocabulary it is part of. Make each Verb **skos:inScheme** of at least one concept scheme (the overarching vocabulary), plus any new versions of the vocabulary with new revisions.

xapi:referencedBy connects an **xapi:Verb** or other xAPI term not directly maintained by the vocabulary author to the **skos:ConceptScheme** that is the xAPI vocabulary referencing the term. Generally speaking, no properties should be provided for referenced Verbs or Activity Types other than **xapi:thirdPartyLabel**, **xapi:referencedBy**, or **skos:scopeNote**. This property (**xapi:referencedBy**) is a sub property of **skos:inScheme**.

Properties for vocabulary provenance metadata

prov:specializationOf comes into play to relate revisions of the same vocabulary. The specific revision of the vocabulary is **prov:specializationOf** for the overarching vocabulary.

prov:wasGeneratedBy connects specific revisions to the ‘provenance activity’ (a **prov:Activity**, unrelated to an xAPI Activity) that generated them. That provenance activity will often be a working group or community of practice, and in those cases it should have at least a **foaf:name**.

prov:wasRevisionOf connects new revisions of the vocabulary to the old revision they’re based on.

Properties for relating terms within a vocabulary

skos:broader goes from the more specific term to the more general. B skos:broader C means that C's definition encompasses B's definition. It is for use inside / within a vocabulary. For example, the IEEE ADB Vocabulary (`https://w3id.org/xapi/adb`) has a verb (highlighted) that is more specific than another verb in the vocabulary (annotated). The **skos:broader** property would be here to here show this relationship. For example:

< `https://w3id.org/xapi/adb/verbs/highlighted` > *skos:broader* < `https://w3id.org/xapi/adb/verbs/annotated` >.

skos:narrower is the inverse of **skos:broader**. This property is also intended for use inside vocabularies.For example:

< `https://w3id.org/xapi/adb/verbs/annotated` > *skos:narrower* < `https://w3id.org/xapi/adb/verbs/highlighted` >.

skos:related leaves very ambiguous the nature of the relationship, but does assert there's some conceptual connection between the two terms it connects. Since it is only for use inside vocabularies, it probably should not be used very often (if a vocabulary is well structured, it usually won't include terms that only have ambiguous relationships to each other). This, narrower, and broader should only be used by the vocabulary author.

Properties for relating terms between different vocabularies

skos:broadMatch is just like **skos:broader** (in fact, it is a sub property) but is intended for use between different vocabularies, and as such is appropriate for use by anyone, which applies to all the ‘Match’ properties.

skos:closeMatch relates a term in a vocabulary to a term in another vocabulary that means very nearly the same thing. While it will be worth considering using the other term directly, it is usually more reasonable to create a new term with the precise meaning desired, then relate it using **skos:closeMatch** to the other vocabulary’s term.

skos:exactMatch makes a bold claim: two concepts are so close that they might reasonably be considered identical. It should only be used after careful consideration. If creating a term for the first time and you discover another term in another vocabulary that would be an exact match, use the other term instead. Only if two vocabularies independently arrived at a term that was nigh identical should **skos:exactMatch** be used. Basically, try very hard not to use this.

skos:narrowMatch is the inverse of **skos:broadMatch**.

skos:relatedMatch is like **skos:related**, but for between different vocabularies. Since two separate vocabularies having ambiguously related terms is normal, it should see use a lot more often than skos:related.

Property for linking a term to a natural language meaning (a “Synset”) in Wordnet

xapi:closelyRelatedNaturalLanguageTerm while this is a sub property of **skos:relatedMatch**, it should only be used by the original vocabulary author. It relates a **skos:Concept**, such as any xAPI Verb, to a closely related **wordnet:Synset** — basically, a natural language meaning. All of the other properties in this section should only be used to relate xAPI terms of the same sort to each other — Verbs to Verbs, Activity Types to Activity Types, etc.

Guidelines for Property Use

The various label properties are special. For any xAPI term with a natural language rendering in-Statement, such as a verb's display, or an activity's name, they're for providing possible forms for that. For all xAPI terms they describe how to refer to them when displaying the vocabulary. For example, even though Activity Types don't have a natural language rendering in statements, the label properties tell us how to write about them in documentation without putting the IRI everywhere.

The note properties, while often including natural language strings that can be used when displaying the vocabulary, also have more flexible uses, such as putting a Statement in the example property. But, like the label properties, they're primarily descriptive; there's little that a computer can do with them that couldn't be done with just an HTML page.

Where we start to see the power of RDF is with the use of SKOS concept schemes, but also the PROV ontology. By relating a specific revision to an overall vocabulary concept scheme and its preceding revision, and then relating each xAPI term to the specific revision and the overall vocabulary, it becomes possible to quickly determine answers to questions like "have any terms been added or removed in this revision?", "is this term in the most recent revision?", and "are there any further revisions beyond revision X?" **Note:** the appropriate way to publish vocabulary datasets under this approach is, when the URL for the overall vocabulary is visited, redirect to the URL for the current revision of the vocabulary.

Add in **prov:wasGeneratedBy**, and it will also be possible to describe things like responsibility shifting for a vocabulary, such as the recent transition of ownership of cmi5 verbs to the ADL vocabulary. See the example below. While these capabilities may seem less important now, look at how many different groups are starting to coin terms already, in these very early days of xAPI. A small amount of extra information now, easily expressed, will make the future much easier. The PROV ontology exists to handle these sorts of provenance tracking in a sane, proven model.

In many ways the most immediately exciting part of using these terms will be the capabilities enabled by the SKOS semantic relationships such as broader, narrower, and so forth. Using these for in- and between-vocabulary relationships of terms will make it possible to answer questions like "are there any similar terms in other vocabularies?", "what are all the verbs I need to query for if I want all the statements this verb applies to as a concept?", "I have this natural language term, are there any xAPI verbs related to it?", and even "If so what are the natural language terms they connect through?".

Example xAPI Vocabulary Dataset (Shown in Turtle RDF)

```
@prefix : <http://myvocab.example.com/xapi/> .
@prefix myvocab10: <http://myvocab.example.com/xapi/1.0/> .
@prefix myvocab20: <http://myvocab.example.com/xapi/2.0/> .
@prefix xapi: <https://w3id.org/xapi/ontology#> .
@prefix adlnet: <https://w3id.org/xapi/adl> .
@prefix skos: <http://www.w3.org/2004/02/skos/core#> .
@prefix prov: <http://www.w3.org/ns/prov#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .

# the above are just bookkeeping for all the prefixes and the current base namespace


# the overall vocabulary, which is the namespace verbs and so forth will be in.
# using just the colon indicates this is just the bare base URI
:
    a skos:ConceptScheme ;
    prov:wasGeneratedBy [
    a prov:Activity ;
        foaf:name "Sports and Competition xAPI Vocabulary Working Group"@en
    ] .


# the current version of the vocabulary, which is connected to the overall vocabulary
# as a specialization, and to the previous version by wasRevisionOf
myvocab20:
    a skos:ConceptScheme ;
    prov:specializationOf : ;
    prov:wasRevisionOf myvocab10: .


# okay, now we're starting to define our verbs. As recommended by SKOS, our concepts
# go in the overall vocabulary namespace, not the version namespaces.
:qualifiedfor
    a xapi:Verb ;
    skos:inScheme : ; # the overall vocabulary again
    skos:inScheme myvocab20: ;
    skos:prefLabel "qualified for"@en ;
    skos:definition "To be accepted as a participant for the Activity object by meetin
g some set of criteria"@en .

:placed
    a xapi:Verb ;
    skos:inScheme : ;
    skos:inScheme myvocab20: ;
    skos:prefLabel "placed"@en ;
    skos:altLabel "ranked"@en ;
```

```
    skos:definition "To achieve a ranked outcome in the Activity object, which is a co
mpetitive event"@en ;
    skos:broadMatch adlnet:completed ;
    xapi:closelyRelatedNaturalLanguageTerm <http://wordnet-rdf.princeton.edu/wn31/2006
61447-v> .

# when we're talking about 'completed' for a race, 'finished' might be a good display
adlnet:completed xapi:thirdPartyLabel "finished"@en .

:medaled
    a xapi:Verb ;
    skos:inScheme : ;
    skos:inScheme myvocab20: ;
    skos:prefLabel "medaled"@en ;
    skos:definition "To receive a medal for achievement in a competitive event"@en ;
    skos:broader :placed .
```

JSON-LD for xAPI Vocabularies

As you can see, the above RDF doesn't look too bad in Turtle (be glad you're spared from the RDF/XML rendering). But at the same time, there's enough complication that things aren't simple. By using JSON-LD, we'll make it so straightforward documents can contain the same information as the above in a way anyone already comfortable with JSON will have no problem with them.

The key to this is the use of a 'context' that describes how different JSON attributes and values get turned into the IRIs used in RDF. JSON-LD pushes complexity into the context, which can be shared across documents, so applications or people writing and reading documents only deal with the simpler JSON representation. Don't worry too much about the details if you're just hearing about JSON-LD. It pretty much is as simple as it looks: the keys provide shorthand for the RDF properties described. When the @type is @id, the values of the key are IRIs, but in other cases they're objects.

While any RDF (HTML/RDFa, Turtle, RDF/XML,etc.) can be expressed in JSON-LD and vice versa, in order to keep complexity low, the use of JSON-LD for xAPI vocabularies is restricted further, to focus on specific structures and properties. That is what makes it easy for producers and consumers to use it with standard JSON tools. This standard JSON-LD practice makes it a wonderful bridge between JSON and the Semantic Web. The more constrained JSON can be safely traversed by simpler, easy to write JSON-only clients, but the use of additional JSON-LD elements and expressivity on top of that structure can communicate arbitrarily rich information. JSON-LD will likely be a good starting point for systems looking to directly retrieve meaning from a number of different vocabularies, when provided the vocabulary IRI.

Example xAPI Vocabulary Dataset (JSON-LD Version)

Some parts of the JSON-LD document, such as those properties using SKOS, are intended for use by semantic web aware processors, while other parts are for general consumption. Except for a few keys being prefixed by “@”, it should look like you might expect a JSON document to look to describe an xAPI vocabulary and associated verbs, even knowing nothing of JSON-LD.

```
{
  "@context": {
  "http://xapi.vocab.pub/vocabulary/context.jsonld",
  "@id": "http://myvocab.example.com/xapi/",
  "@type": "ConceptScheme",
  "versions": [
    {
      "@id": "http://myvocab.example.com/xapi/2.0/",
      "wasRevisionOf": "http://myvocab.example.com/xapi/1.0/"
    },
    {
      "@id": "http://myvocab.example.com/xapi/1.0/"
    }
  ],
  "wasGeneratedBy": {
    "name": {
      "en": "Sports and Competition xAPI Vocabulary Working Group"
    }
  },
  "verbs": [
    {
      "@id": "http://myvocab.example.com/xapi/qualifiedfor",
      "@type": "Verb",
      "skos:inScheme": {
        "@id": "http://myvocab.example.com/xapi/2.0/"
      },
      "prefLabel": {
        "en": "qualified for"
      },
      "definition": {
        "en": "To be accepted as a participant for the Activity object by meeting some
 set of criteria"
      }
    },
    {
      "@id": "http://myvocab.example.com/xapi/placed",
      "@type": "Verb",
      "skos:inScheme": {
```

```
        "@id": "http://myvocab.example.com/xapi/2.0/"
      },
      "prefLabel": {
        "en": "placed"
      },
      "altLabel": {
        "en": "ranked"
      },
      "definition": {
        "en": "To achieve a ranked outcome in the Activity object, which is a competit
ive event"
      },
      "broadMatch": [
        "http://www.adlnet.gov/expapi/verbs/completed"
      ],
      "closelyRelatedNaturalLanguageTerm": "http://wordnet-rdf.princeton.edu/wn31/2006
61447-v"
    },
    {
      "@id": "http://myvocab.example.com/xapi/medaled",
      "@type": "Verb",
      "skos:inScheme": {
        "@id": "http://myvocab.example.com/xapi/2.0/"
      },
      "prefLabel": {
        "en": "medaled"
      },
      "definition": {
        "en": "To receive a medal for achievement in a competitive event"
      },
      "broader": [
        "http://myvocab.example.com/xapi/placed"
      ]
    }
  ]
}
```

Additional Tips for JSON-LD Documents

By using the following set of guidelines, published vocabulary documents will be more usable by both semantic web and simpler JSON-using clients. In these rules, a “complete IRI” is an in-document IRI that doesn’t use any of the shorthand prefix options JSON-LD makes possible.

1. The @id of each individual term and vocabulary in the document must be a complete IRI. Refer to the example above. Notice that both the vocabulary IRI (`https://w3id.org/xapi/adl`) and “abandoned” verb IRI (`https://w3id.org/xapi/adl/verbs/abandoned`) use a complete IRI for the identifier.
2. Don’t use **@graph** for grouping Verbs or Activity Types into a vocabulary dataset. Instead use the value the type of vocabulary dataset being represented (e.g., “verbs” or “activity-types”). See the previous JSON-LD code sample above in this document for an example.
3. The value of “verbs” in a vocabulary is an array of verbs, with complete IRIs for **@id**. Note: **@type** is needed here, as it is not inferred.
4. For all of **skos:broader**, **skos:narrower**, **skos:related**, **skos:closeMatch**, **skos:exactMatch**, **skos:broadMatch**, **skos:narrowMatch**, and **skos:relatedMatch**, the value must be an array of complete IRIs.
5. For all of **skos:definition**, **skos:prefLabel**, **skos:altLabel**, **xapi:thirdPartyLabel**, **foaf:name**, the value must be a language map (like in xAPI!). Note that name is not for naming things generally — it’s for people, organizations, groups, etc. The usual construct for naming xAPI’s concepts is prefLabel.
6. **xapi:closelyRelatedNaturalLanguageTerm**’s value must be a complete IRI (not in an array).
7. **prov:wasGeneratedBy** should be used on the root vocabulary, or more rarely on a vocabulary that is a specialization of the root vocabulary, with a value that is an object (and has a name), indicating the organization or other group involved in its creation.
8. If you’re serving up something that isn’t a vocabulary, but is part of one, use **vocabulary** with a value that is the overall vocabulary IRI.

Again, these guidelines are for generating JSON-LD that can be used by both simple JSON clients and more complex clients. Not all JSON-LD generated for xAPI-related applications will need to follow these rules to be useful. Clients based on the above rules should pivot on the presence of **@type** matching “Verb” or “Activity Type”, and servers should avoid those values for the **@type** element (just use **xapi:Verb**, **xapi:ActivityType**, **skos:ConceptScheme**, etc.) when the JSON-LD they’re producing won’t be readily consumable per the above rules.

Search and Query Opportunities

A brief overview of vocabulary search and reuse use is presented in the xAPI Companion Specification. Publishing vocabularies as linked datasets opens up several opportunities for search and reuse. By exposing vocabularies as HTML and RDF it makes them inherently more discoverable, even in public search engines such as Google. Access and discoverability through public search engines can be also controlled and restricted on the server where vocabularies are published or hosted.

Direct Requests on Vocabulary Dataset IRIs

One of the most common ways people or applications can obtain more meaning is by directly targeting the vocabulary dataset IRI. This method is useful for applications that might require immediate need to access to a whole vocabulary dataset, meaning of a single term, or the relationships among terms in vocabulary. Let's assume that you are building an authoring tool/application and want to find out all of the published ADL vocabulary. You already know that you can directly view these in a browser at the vocabulary IRI. For example, the overall ADL Vocabulary is available here: ***https://w3id.org/xapi/adl***. But if you wanted to retrieve a machine-readable representation of the vocabulary dataset for your application and update the displays of this information on a regular basis you could simply issue a HTTP request. The example below a sample request using the CURL command line tool. The request is for the ADL vocabulary in JSON.

```
curl --header "Accept: application/json" -L https://w3id.org/xapi/adl
```

Alternatively, you could use JavaScript to test that JSON is returned using an HTTP request in a browser developer console such as Firefox or Chrome:

```
<!doctype html>
<html>
  <head>
    <script src="http://code.jquery.com/jquery-1.4.2.min.js"></script>
    <script type="text/javascript">
      $(document).ready(function() {
        $.ajax({
          url: 'https://w3id.org/xapi/cmi5',
          type: 'GET',
          dataType: 'json',
          beforeSend: setHeader
        });
      });

      function setHeader(xhr) {
        xhr.addEventListener('load', function (ev) {console.log(ev);console.log(xhr.re
sponse);console.log(xhr.responseText);});
        xhr.setRequestHeader('Accept', 'application/json');
      }
    </script>
  </head>
  <body>
    <h1>Load this page with the browser developer console open and view the network /
xhr requests.</h1>
  </body>
</html>
```

It is expected that more vocabularies will be made available as Communities of Practice publishe them. In the meantime, direct GET requests on vocabulary IRIs can be attempted on any of the following vocabularies:

- ADL Vocabulary: `https://w3id.org/xapi/adl`
- IEEE ADB Vocabulary: `https://w3id.org/xapi/adb`
- Across X Vocabulary: `https://w3id.org/xapi/acrossx`
- Serious Games Vocabulary: `https://w3id.org/xapi/seriousgames`
- Video Vocabulary: `https://w3id.org/xapi/video`
- SCORM Vocabulary: `https://w3id.org/xapi/scorm`
- cmi5 Vocabulary: `https://w3id.org/xapi/cmi5`

In addition, valuable information and meaning can be further retrieved from each vocabulary term. The vocabulary datasets use SKOS to express relationships between terms and the vocabulary they belong to (the concept scheme). However, the vocabulary datasets also provide text labels in other languages, descriptions, and links to other sources such as Wordnet for word sense disambiguation. In fact, if you take advantage of “following your nose” to look up other linked data sources, then you will find that each verb should have a link to a wordnet synset. In the previous examples a vocabulary dataset IRI was requested and JSON-LD was returned. The IRI `http://wordnet-rdf.princeton.edu/wn31/200485097-v` is provided as the ***xapi:closelyRelatedNaturalLanguageTerm*** for ADL’s “completed” Verb. A subsequent HTTP request on this IRI returns even more useful data, including multiple language translations and other synset members such as the equivalent word “finished,” `http://wordnet-rdf.princeton.edu/wn31/finish-v` .

```
curl --header "Accept: application/ld+json" -L http://wordnet-rdf.princeton.edu/wn31/200485097-v
```

SPARQL Queries Made Possible

Back to why we're doing all of this: for what it makes possible. There isn't just one thing, but a key component is the ability to answer those first example questions provided in the introduction of this document. Here are some ways SPARQL could be used to do that. While not exactly self explanatory, these should be clear enough to see how the use of the terms above make such retrievals easy (and easy to hide behind a web application with good user experience). Federated semantic queries will also be an option in the future, but at the time of this publication only the ADL vocabulary server is currently supporting RDF in xAPI implementations.

The following example SPARQL queries assumes the reader has some knowledge of SQL-like queries and access to a public SPARQL endpoint. For these examples, the xAPI vocabulary server has provided a public SPARQL endpoint, ***http://xapi.vocab.pub/sparql***.

1.**Has anyone published a translation of any existing xAPI verbs from other vocabulary authors in Simplified Chinese (or any other language)?**

```
PREFIX xapi: <https://w3id.org/xapi/ontology#>

select distinct ?verb ?thirdPartyLabel

where {
   ?verb a xapi:Verb .
  ?verb xapi:thirdPartyLabel ?thirdPartyLabel .
  FILTER(langMatches(lang(?thirdPartyLabel), "zh-tw"))
}
```

Using the above SPARQL query at the endpoint provided (***http://xapi.vocab.pub/sparql***) will return the following results:

verb	thirdPartyLabel
http://adlnet.gov/expapi/verbs/answered	"回答"@zh-tw
http://adlnet.gov/expapi/verbs/asked	"提問"@zh-tw
http://adlnet.gov/expapi/verbs/attempted	"開始嘗試"@zh-tw
http://adlnet.gov/expapi/verbs/commented	"回應"@zh-tw
http://adlnet.gov/expapi/verbs/completed	"完成"@zh-tw
http://adlnet.gov/expapi/verbs/interacted	"互動"@zh-tw
http://adlnet.gov/expapi/verbs/shared	"分享"@zh-tw
https://w3id.org/xapi/adb/verbs/bookmarked	"加書籤"@zh-tw
https://w3id.org/xapi/adb/verbs/highlighted	"畫重點"@zh-tw
https://w3id.org/xapi/adb/verbs/noted	"做筆記"@zh-tw
https://w3id.org/xapi/adb/verbs/read	"閱讀"@zh-tw
https://w3id.org/xapi/adb/verbs/referenced	"查詢參考"@zh-tw
https://w3id.org/xapi/adb/verbs/requested	"求助"@zh-tw
https://w3id.org/xapi/adl/verbs/logged-in	"登入"@zh-tw

Chinese (Simplified (zh-cn) and Traditional (zh-tw)) and English (en) translations have been provided at the time of this publication. It is expected that more will follow with the adoption of these vocabulary practices. In the future, you could easily substitute the “zh-tw” country-locale code for any other to match your inquiry. **Note:** The xAPI Chinese Community of Practice published translations of pre-existing xAPI verbs that they used as part of their vocabulary dataset at the IRI, `https://w3id.org/xapi/acrossx`. They used the **xapi:thirdPartyLabel** property for translating xAPI verbs from other authors and they used **skos:prefLabel** for translating any original vocabulary terms.

For example, to see only the original verbs that were translated in Simplified Chinese we would simply use the following query:

```
PREFIX xapi: <https://w3id.org/xapi/ontology#>

select distinct ?verb ?label

where {
  ?verb a xapi:Verb .
  ?verb skos:prefLabel ?label .
  FILTER(langMatches(lang(?label), "zh-cn"))
}
```

2.How many other published xAPI vocabularies are reusing/referencing an existing Verb?

```
# return all of the verbs that are referenced by other vocabularies, and show the labe
l, description, scope note, and third party labels

PREFIX xapi: <https://w3id.org/xapi/ontology#>

select distinct ?verb ?label ?description ?vocab ?note ?thirdPartyLabel

where {
{ ?verb a xapi:Verb .
  ?verb skos:prefLabel ?label .
  ?verb skos:definition ?description . }

{ GRAPH
 ?graph { ?verb xapi:referencedBy ?vocab .
       OPTIONAL { ?verb skos:scopeNote ?note }
       OPTIONAL { ?verb xapi:thirdPartyLabel ?thirdPartyLabel }
       }
   }
}
ORDER BY ?verb
```

Below is a SPARQL Query to locate which Activity Types are reused/referenced by other vocabularies. The below query will show the English label and description, the vocabulary referencing the Activity Type, scope note, and thirdPartyLabel. Paste the following query into the ***http://xapi.vocab.pub/sparql*** endpoint or ***click here*** for the results.

```
PREFIX xapi: <https://w3id.org/xapi/ontology#>

select distinct ?activity ?label ?description ?vocab ?note ?thirdPartyLabel

where {
{ ?activity a xapi:ActivityType .
  ?activity skos:prefLabel ?label .
  ?activity skos:definition ?description .}

{ GRAPH
 ?graph { ?activity xapi:referencedBy ?vocab .
       OPTIONAL { ?activity skos:scopeNote ?note }
       OPTIONAL { ?activity xapi:thirdPartyLabel ?thirdPartyLabel }
       }
   }
}
ORDER BY ?activity
```

The previous two examples show all of the results of terms that are referenced by other vocabularies. If you wanted to be more specific and narrow down the results to a specific verb (e.g., ADL's "completed" verb), you could perform the query below. Use the ***http://xapi.vocab.pub/sparql*** endpoint or ***click here*** for the results. This example query also displays other properties such as **xapi:closelyRelatedNaturalLanuageTerm**.

```
PREFIX adl: <http://adlnet.gov/expapi/verbs/>
PREFIX xapi: <https://w3id.org/xapi/ontology#>

select distinct ?label ?description ?synset ?vocab ?note ?thirdPartyLabel

where {
  {
    adl:completed skos:prefLabel ?label .
    adl:completed skos:definition ?description .
    adl:completed xapi:closelyRelatedNaturalLanguageTerm ?synset .
  }
{
GRAPH ?graph {
    adl:completed xapi:referencedBy ?vocab .
    OPTIONAL { adl:completed skos:scopeNote ?note }
    OPTIONAL { adl:completed xapi:thirdPartyLabel ?thirdPartyLabel }
    }
  }
}
```

3.**What verbs are more specialized forms of other verbs?**

```
PREFIX xapi: <https://w3id.org/xapi/ontology#>
PREFIX skos: <http://www.w3.org/2004/02/skos/core#>

SELECT distinct ?verb ?verbLabel ?verbDescription ?broaderVerb ?broaderLabel ?broaderD
escription

 WHERE { {
    ?verb a xapi:Verb .
    ?verb skos:prefLabel ?verbLabel .
    ?verb skos:definition ?verbDescription .
    ?verb skos:broader ?broaderVerb }
{
GRAPH ?graph {
  OPTIONAL { ?broaderVerb skos:prefLabel ?broaderLabel }
  OPTIONAL { ?broaderVerb skos:definition ?broaderDescription }
  }
 }
}
```

Note: There might be limited results in the above example as at the time of this writing only the IEEE ADB Verb vocabulary had applied any form of semantic relationship mapping between terms both inside and outside of the ADB vocabulary. It is expected that vocabulary tools and refined publishing processes will help make this practice more practical and widespread in the near future.

Acknowledgements

The development of the vocabulary dataset schema was facilitated by the Advanced Distributed Learning (ADL) Initiative and developed by the xAPI Vocabulary Working Group. This document was originally authored by Mr. Russell Duhon.