Navy Core xAPI Profile

Core Requirements and Guidelines for Implementing xAPI

Document Version: 1.2

Table of Contents

Revision History

Version	Release Date	Notes
0.5	5/05/2020	Initial draft of the document. Authors review only.
0.9	6/29/2020	Second draft of the document. Internal Navy review only.
1.0	7/30/2020	All edits and feedback incorporated. First full version of the document released.
1.1	7/20/2021	Global update from “NETC” to “Navy.”
1.2	5/20/2022	Clarification of requirements and supporting text. Typos and copy/paste fixes.

Table 1: Navy Core xAPI Profile Revision History

1. Overview

1.1. Purpose

The Navy Core xAPI Profile defines the requirements for all Navy xAPI implementations regardless of content type or other tracking requirements. This profile includes information that, after successful implementation, results in an interoperable data environment. All of the published xAPI profiles inherit requirements from the Navy Core xAPI Profile. All community-extended profiles used by the Navy should also inherit requirements from the Navy Core xAPI Profile.

1.2. How to Read this Document

This document should be interpreted as the authoritative Navy guidance on implementing xAPI in the Navy. This document is provided as an auxiliary resource to the formal xAPI Specification (version 1.0.3) and provides Navy-specific requirements. If an object or property is listed as "optional" in the xAPI Specification, but that same object or property is listed as a requirement herein, then this document takes precedence.

The xAPI Implementation Guidelines section of this document provides content developers with the mandatory obligations for each part of the xAPI Statement Data Model. Additional requirements for xAPI Statements are further defined in other Navy xAPI Profiles. The xAPI Content Integration Requirements Section provides content developers with broader technical guidance on integrating and launching content. The requirements provided in this document should be followed first before implementing any additional requirements provided in other Navy xAPI Profiles.

The following common syntactic and typographic conventions will be used in this document:

- Footnotes appear at the bottom of pages where there is a need to provide additional information or references.
- Words or concepts will use *Italics* or "double quotations" when there is a need to draw special attention to them.
- Title Case is used to identify concepts and terminology from the xAPI Specification.
- A `Courier font such as this` is used to represent data objects or properties associated with the xAPI Statement Data Model. A `Courier font` is also used to distinguish between technical terminology or concepts originating from the xAPI Specification or an xAPI Profile. A `Courier font` is also used to represent unique identifiers such as IRIs or IDs.
- xAPI Statement examples represented as JSON snippets and other types of syntax examples are provided inside of light blue background box area like the one below:

```
"data-object":{
        "property1": "value",
        "property2": "value"
}
```

1.2.1. Conformance Requirements

There are two levels of obligation with regards to xAPI conformance requirements identified by the terms herein. Since these xAPI conformance requirements are intended to be reused as part of the contract requirements in a Data Item Description (DID) deliverable or Contract Data Requirements List (CDRL), the use of "SHOULD" and "MAY" was avoided since these types of contract documents normally contain only mandatory instructions. Complete definitions are provided below. Not following these recommendations could risk interoperability and and/or lead to xAPI deliverables not being accepted by the Navy.

1. SHALL OR SHALL NOT. If a product fails to implement a SHALL or SHALL NOT requirement, the product is non-conformant to the Navy's requirements.

2. xAPI Implementation Guidelines

2.1 xAPI Statement Requirements

This section will primarily address the core requirements associated with the Statement Resource and the xAPI Statement Data Model. The xAPI Statement Data Model provides content developers with the ability to represent learning experience and performance data in a structured manner. Statements are modeled using JSON Objects[1], and are fundamentally expressed in the form "Actor, Verb, Object" or "A Person(s) did something." Statements also typically include additional information in the Result and Context Objects to add more meaning or details about the learning experience. A high-level diagram of the primary parts of the xAPI Statement Model is provided below.

[1] See https://www.json.org for more information.

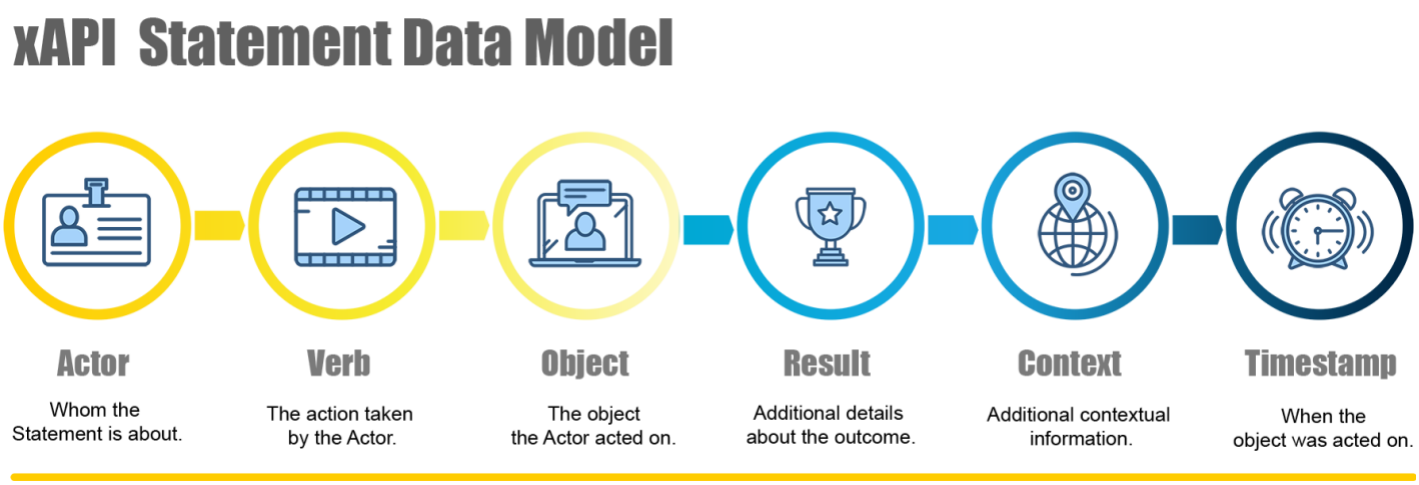

Figure 1:High-level Parts of the xAPI Statement Data Model

2.1.1. Actor

For Navy xAPI implementations, Actors will represent the individual person (Actor) who performed the action (Verb) in a given Statement. When there is a scenario identified with the need to capture information about a group of learners, a future Navy xAPI Profile will further define and describe the usage of a Group in an xAPI Statement.

2.1.1.1. Actor Requirements

The following are the requirements for an Actor in an xAPI Statement:

- The `actor.name` SHALL be set to the full name of the Actor in the format "First Last".
- The `actor.objectType` SHALL be set to `Agent` unless defined differently in a specific Navy xAPI Profile.
- The `actor.account.homepage` SHALL be set to `https://edipi.navy.mil`
- The `actor.account.name` SHALL be set to the EDIPI associated with the user.

In most cases when the content is web-based or otherwise launched from the LMS, the Actor will be automatically provided to the content during launch. In these cases, the requirements above will be fully handled by the launching system and not directly managed by the developer. See Section 2.2 for more information on launching content from the LMS or outside the LMS.

2.1.1.2. Actor Example

```
"actor":{
       "name": "John Doe",
       "objectType": "Agent",
       "account":{
           "name":"0123456789",
           "homepage":"https://edipi.navy.mil"
       }
}
```

2.1.2. Verbs

A Verb conveys the action that occurred in an xAPI Statement. The Verb in an xAPI Statement is represented in the past tense since the Statement is triggered immediately after a learning experience or event occurs. A list of Verbs are provided in each of the Navy xAPI Profiles. See the general xAPI Statement requirements for Verbs in the following section.

2.1.2.1. Verb Requirements

The following are the requirements for a Verb in an xAPI Statement:

- The `verb.display.en` SHALL be set to the human-readable, past tense representation of the Verb.

2.1.2.2. Verb Example

```
      "verb":{
            "id":"http://adlnet.gov/expapi/verbs/initialized",
            "display":{
                  "en":"initialized"
            }
```

```
        }
```

2.1.3. Objects

The Object of an xAPI Statement defines the thing that was acted on. The Object can be an Activity, Agent/Group, SubStatement, or Statement Reference. By default, Objects will be an Activity unless otherwise specified in a particular Navy xAPI Profile. The specific xAPI Statement requirements for Activities are provided in the following section.

2.1.3.1. Activity Object Requirements

The following are the requirements for an Activity Object in an xAPI Statement:

- The Activity ID (`object.id`) SHALL be a unique identifier based on the requirements for Activity IDs in Section 2.1.3.2.
- The Activity Object SHALL only have one Activity ID to represent it.
- The `object.definition.name.en` SHALL be set to the official name or title of the Activity.
- The `object.definition.description.en` SHALL be set to a short description of the Activity.
- The `object.definition.type` SHALL be set to the identifier associated with the relevant Activity Type. Valid Activity Types are included in the respective Navy xAPI Profiles.

2.1.3.2. Activity ID Requirements

Activity IDs and Objects have a direct one-to-one relationship. Each Activity ID uniquely identifies the Object. Ensure that there is absolutely no possibility of accidentally creating and using the same Activity IDs for different activities. Do not use multiple Activity IDs to represent the same Object and do not reuse the same Activity ID to represent a different Object.

The following are the requirements for an Activity ID for an Object in an xAPI Statement:

- The Activity ID SHALL uniquely identify the Object.
- The Activity ID SHALL NOT be used to represent any other Object.

2.1.3.3. Activity Object Examples

The following examples are provided for illustrative purposes and do not include all possible types of activities.

2.1.3.3.1. Course Activity Example

```
"object":{
  "id": "https://navy.mil/netc/xapi/activities/courses/37823a7a-afee-42aa-c4ee-3333acac402",
  "objectType": "Activity",
  "definition": {
    "name": {
      "en": "P-8A Poseidon Course"
    },
    "description": {
      "en": "An interactive e-learning course on the Navy's P-8A aircraft."
    },
    "type": "http://adlnet.gov/expapi/activities/course"
  }
}
```

2.1.3.3.2. Lesson Activity Example

```
"object":{
  "id": "https://navy.mil/netc/xapi/activities/lessons/9e32f474-af07-11ea-b3de-0242ac130004",
  "objectType": "Activity",
  "definition": {
    "name": {
      "en": "Advanced Airborne Sensor Lesson"
    },
    "description": {
      "en": "A lesson on advanced airborne sensor general familiarization from the P-8A course."
```

```
    },
    "type": "http://adlnet.gov/expapi/activities/lesson"
  }
}
```

2.1.3.3.2. File Activity Example

```
"object":{
  "id": "https://navy.mil/netc/xapi/activities/files/44662c4a-a764-11ea-bb37-
0242ac130002",
  "objectType": "Activity",
  "definition": {
    "name": {
      "en": "Advanced Airborne Sensor Special Mission Pod Deployment Mechanism
Technical Manual"
    },
    "description": {
      "en": "A tech pub PDF used with training on advanced airborne sensor general
familiarization."
    },
    "type": "http://adlnet.gov/expapi/activities/file"
  }
}
```

2.1.3.4 Activity Object Extensions

When used as part of Activity Object Definition, the Extensions Object provides a way to optionally extend xAPI Statements to include additional information about the activity (object). The values of Extensions can be any value or JSON Object. Refer to the Navy Common Reference Profile and other Navy Profiles for existing extension requirements and examples. Additional Extensions may be added in future versions of the Navy Profiles.

2.1.4. Context

The Context of an xAPI Statement contains additional information related to a learning experience or event. Depending on the Navy xAPI Profile, the information in the Context Object of an xAPI Statement may vary. Please see the general xAPI Statement requirements for examples of these objects and their properties in the sections below. More specific Context examples are further defined in each of the respective Navy xAPI Profiles.

2.1.4.1 Context Requirements

The following are the requirements for the Context in an xAPI Statement:

- The `context.contextActivities.category` array SHALL contain the Navy xAPI Profile activity.
    - Navy xAPI Profile activity:

```
{
   "id": "https://w3id.org/xapi/netc/v1.0",
   "definition": {
      "type":"http://adlnet.gov/expapi/activities/profile"
   }
}
```

- Profile Activities for each Profile SHALL be declared in the `category` array for each Navy xAPI Profile that is applied to an xAPI Statement. Please refer to the respective Navy xAPI Profiles for the Context Activities Category requirements.
- If the object of the statement is a child of some part of a hierarchy, its direct parent, as an xAPI Activity, SHALL be added to the `context.contextActivities.parent` array at index 0,
    - If the direct parent is conceptual and will never be the object of a statement, the next parent up the hierarchy chain SHALL be added to the `context.contextActivities.parent` array at index 1
    - Parent activities SHALL be added to the hierarchy chain until either the entire parent hierarchy is represented, or an activity in the array is known to be the object of other xAPI Statements
- The value of the `context.registration` property SHALL be a Universally Unique Identifier (UUID)
- The value of the `context.registration` property SHALL be set to a unique UUID for each attempt

- The value of the `context.platform` property SHALL be set to a text value, provided by Navy Learning Stack Administrator

2.1.4.2 Context Activities

The Context Activities property is used to identify activities that are related to the xAPI Statement. This could be other activities in the same course (`grouping`), the parent activity of the activity in the current xAPI Statement (`parent`), an activity representing some general attribute of the xAPI Statement (`category`), or any other activity that has some relationship to the activity or the xAPI Statement (`other`). The requirements for using `context.contextActivities.category` and `context.contextActivities.parent` properties are addressed in this document. Please refer to the use case specific Navy xAPI Profiles for additional requirements for the other `context.contextActivities` properties (i.e., `grouping` and `other`).

2.1.4.2.1 Context Activities Parent

Typically learning content is structured into hierarchical structures using concepts such as courses, modules, or lessons. And these are related to each other to form the structure of a course. A lesson, for example, may cover a specific topic and that topic may be one of many in a course. This hierarchical structure is often how the content is organized in LMSs or players of the content, it is how learners see and experience the content, and it is how content was originally created. But this structure is not known intrinsically by the LRS, which can make visualizing, querying, and organizing data difficult. However, xAPI Statements can contain some additional context, like the parent activity of the current object. This additional context allows the LRS to build an internal representation of the course hierarchy.

The Context Activities Parent array is a place to list activities that are the parents of the object of the xAPI Statement. For example, if the object of the statement is a lesson, the activity in the Context Activities Parent array would represent that lesson's immediate parent, such as a module or course. Another example is a video in a lesson. The statements about the video would include the lesson activity in the Context Activities Parent array. Since the Context Activities Parent property is an array it can contain more than one parent activity. The LRS will build the hierarchy of activities based on the order in which they are listed in that array. This *Chain of Parents* gives the Learning Record Providers a way to represent more of the content hierarchy tree than just the immediate parent.

The requirements for the Context Activities Parent property are included in the list of requirements for the statement Context property in Section 2.1.4.1. An example is provided below showing how to use the Context property to demonstrate the chain of parents. In this example, the object of the Statement is assumed to be an assessment question (cmi.interaction) in the hierarchy where the direct parent (Deep Learning Assessment) is conceptual and will not itself be the object of a statement.

```
"context":{
  "contextActivities":{
    "parent":[
      {
        "id":"https://www.coursera.org/dl/assessment/01",
        "definition":{
          "name":{
            "en":"Deep Learning Assessment"
          },
          "type":"http://adlnet.gov/expapi/activities/assessment"
        }
      },
      {
        "id":"https://www.coursera.org/specializations/deep-learning/1",
        "definition":{
          "name":{
            "en-US":"Neural Networks and Deep Learning"
          },
          "description":{
            "en-US":"Neural Networks and Deep Learning"
          },
          "type":"http://adlnet.gov/expapi/activities/lesson"
        }
      }
    ]
  }
}
```

2.1.4.3 Context Registration

The Registration property is used to identify multiple xAPI Statements that are all part of a particular attempt. The value of the Registration property should persist throughout all Statements during each attempt. Please refer to the respective Navy xAPI Profiles for more specific requirements and usage of Registration.

2.1.4.4 Context Extensions

When used as part of Context, the Extensions Object provides a way to optionally extend xAPI Statements to include additional information about the core learning experience that the Statement is representing. The values of Extensions can be any value or JSON Object. Refer to the Navy Common Reference Profile and other Navy Profiles for existing extension requirements and examples. Additional Extensions may be added in future versions of the Navy Profiles.

2.1.4.5 Context Platform

The `context.platform` property is used to specify the computer system's software or hardware used while the Actor experienced the content. xAPI Statements are required to include the `context.platform` property if the value is known. The value of the `context.platform` property will vary depending upon the Navy Profile used. For example, the `context.platform` property in the Navy E-learning Profile includes the LMS and its version, which is determined by a Navy Learning Stack Administrator. Similarly, the `context.platform` property for a mobile performance support app will display the version of the app. Since this value may vary depending on the type of content, the content developer is expected to contact Navy learning stack administrators to verify the correct value.

2.1.4.6. Context Example

```
"context": {
  "contextActivities": {
    "category": [ {
      "id": "https://w3id.org/xapi/netc/v1.0",
      "definition": {
        "type": "http://adlnet.gov/expapi/activities/profile"
      }
    }]
  },
  "registration": "d7bbe8a0-6c77-41a0-a230-9f70d9df9204",
  "platform": "Moodle 3.8.3"
}
```

2.1.5. Result Object

The Result of an xAPI Statement contains additional information related to a measured outcome. Depending on the Navy xAPI Profile, the information in the Result Object of an xAPI Statement may vary. Specific Result requirements and examples are further defined in each of the respective Navy xAPI Profiles.

2.1.5.1 Result Object Extensions

When used as part of the Result Object, the Extensions Object provides a way to optionally extend xAPI Statements to include additional information about the outcome of the learning experience. The values of Extensions can be any value or JSON Object. Refer to the Navy Profiles for existing extension requirements and examples. Additional Extensions may be added in future versions of the Navy Profiles.

2.1.6. Timestamps

The `timestamp` property is used to provide the time when a learning experience occurred. All Statements SHALL include a Timestamp. A Timestamp SHALL be formatted according to the RFC 3339[2]. This format uses the Gregorian Calendar and Coordinated Universal Time (UTC). The "Z" suffix denotes a UTC offset of 00:00, which is known as Zulu time. Two examples are provided below:

2020-04-30T23:20:50Z

This example represents 20 minutes and 50 seconds after the 23rd hour of April 30th, 2020 in UTC.

2020-06-27T12:55:32-0500

This example represents 55 minutes and 32 seconds after the 12th hour of June 27th, 2020 in the Eastern Time Zone (GMT -5).

2.1.6.1 Requirements

The following are the requirements for a Timestamp in an xAPI Statement:

- The `timestamp` value SHALL be set to the date/time of when the event occurred and not a future time.
- The `timestamp` value SHALL be formatted according to RFC 3339.

[2] "RFC 3339 - Date and Time on the Internet: Timestamps" - A profile of ISO 8601 normal format. https://www.ietf.org/rfc/rfc3339.txt. Accessed 21 Apr. 2020.

- The `timestamp` value SHALL be formatted using the Gregorian Calendar with a time zone offset specified.

2.1.6.2. Timestamp Example

```
"timestamp":"2020-04-30T23:20:50.52Z"
```

2.2. xAPI Content Integration Requirements

While the previous sections of this document have primarily focused on the data requirements for xAPI, this section covers the xAPI content development and integration requirements. The topic of content integration includes how content should be planned for development and configured for connecting and storing data in the Navy LRS. It also includes considerations for how xAPI content will be launched. Refer to the following subsections for more specific content integration requirements.

2.2.1. Launching Content in the LMS

When launching content from the LMS, the LMS will be configured to perform launch using "TinCan Launch." In this case, the content does not need to form the xAPI Actor or hardcode the LRS credentials. This information is provided to the content as part of launch. The content SHALL use the information provided via launch over any locally configured information.

The following query string is an example of the launch information provided as part of launch:

```
http://localhost:8080/run.html
   ?endpoint=https://lrs.navy.mil/xAPI/
   &auth=Basic dG9tOjEyMzQ=
   &Actor={"name":"Valerie Verifierwicz",
   "objectType":"Agent","account":{"homePage":"https://edipi.navy.mil","name":"0123456789"}
```

The parameters content can expect to receive are:

- **endpoint** - The URL to the LRS's xAPI endpoint (e.g. https://lrs.navy.mil/xAPI/)
- **auth** - A limited scope, one time use value of the authorization header used to access to the xAPI Endpoint (e.g. Basic dG9tOjEyMzQ=)

- **actor** - The JSON representation of the Actor Object is required in every xAPI Statement sent by LRP. In this document, the information returned from the LRS is referenced in this document as the learner Agent Object. (e.g. {"name":"Valerie Verifierwicz", "objectType":"Agent","account":{"homePage":"https://edipi.navy.mil","name":"0123456789"})

- **activity_id** - The developer SHALL NOT use the IRI of the Activity (e.g. https://myserver.com/course/introtoxapi101a/lesson1) returned from the launch server and instead use the actual Activity IDs for the content. This assumes a single activity is launched. However, for the Navy, sub-activities are also tracked so there is not a direct correlation between this value and the data being tracked.
- **registration** - The developer SHALL NOT use the registration UUID provided by the launch server. Some Navy Profiles contain attempt and session rules that are not available to the launch service. As a result, the Registration value supplied during launch is not valid. The developer may choose to use the Registration value only for new Initialize Statements that begin a new attempt.

2.2.2. Launching Outside the LMS

For content that is not web-based or situations where there is a need to have the Actor's identity anonymized (e.g., BYOD mobile devices), the Navy LRS Administrator WILL provide content developers with specific LRS Endpoints and authentication requirements for launching, testing, and deploying xAPI content in the various Navy environments (e.g., development, production, etc.).

If you are developing xAPI content that is not web-based or will not be launched from the LMS, contact the Navy Learning Stack Administrator for information on authenticating with the AAS. After authenticating with the AAS, the developer is responsible for forming the xAPI Actor as defined in Section 2.1.1.2.

2.2.3. Recording Attempt Information in xAPI Statements

The Navy profiles require attempts to be identified by a UUID. This UUID value is stored in the Context Registration property of the xAPI statement. See the Context Registration Section of this document for more information.

2.2.3.1. Context Extension Activity Attempt Map

The context registration property only contains the attempt UUID for the activity of the statement. The attempt information for the parent activities and objects higher up the hierarchy is not retained in that statement. The activity attempt map extension was developed to resolve this loss of attempt information. It holds a mapping of activity IRIs and attempt UUIDs for the activities up the hierarchy from the current activity.

```
context: {
      extensions: {
            "https://navy.mil/extension/activity-attempt-map": {
                  "https://navy.mil/xapi/activity/092333": "f4d6e06d-8d06-4819-
8b4a-cc1f48a1a575",
                  "https://navy.mil/xapi/course/aabbc": "f3d2b881-304b-4ca3-b5cf-
152b0b3539d8"
            }
      }
}
```