cmi5 Best Practices Guide:

From Conception to Conformance

Rustici Software

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
06-14-2021	Document	09/21/2020- 09/20/2021

4. TITLE AND SUBTITLE

cmi5 Best Practices Guide: From Conception to Conformance

5a. CONTRACT NUMBER

W900kk1890005

5b. GRANT NUMBER

N/A

5c. PROGRAM ELEMENT NUMBER

0603769D8Z

6. AUTHOR(S)

Brian Miller, Rustici Software
Tammy Rutherford, Rustici Software
Alicia Pack, Rustici Software
George Vilches, Rustici Software
Jim Ingram, Rustici Software

5d. PROJECT NUMBER

N/A

5e. TASK NUMBER

N/A

5f. WORK UNIT NUMBER

N/A

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Rustici Software LLC
210 Gothic Ct #100
Franklin, TN 37067

8. PERFORMING ORGANIZATION REPORT NUMBER

N/A

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Defense Support Services Center
Advanced Distributed Learning Initiative
13501 Ingenuity Drive, Suite 248
Orlando, Florida 32826

10. SPONSOR/MONITOR'S ACRONYM(S)

OUSD/P&R/DSSC/ADLI

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

Distribution A

13. SUPPLEMENTARY NOTES

14. ABSTRACT

cmi5 is an xAPI Profile that provides a set of rules for how online courses are imported, launched, and tracked using a Learning Management System (LMS) and xAPI, which is a key piece missing from the xAPI data standard. cmi5 uses a simplified tracking model specifying only the most essential elements for interoperability across most learning instances, including score, status, and time. While cmi5 only explicitly defines the necessities, it's capable of recording and reporting on much more.

15. SUBJECT TERMS

cmi5, xAPI Profile

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Sae Schatz
U	U	U	UU	35	19b. TELEPHONE NUMBER *(Include area code)* 757-572-6206

Acknowledgements

Principal Authors

Brian Miller, Rustici Software
Tammy Rutherford, Rustici Software
Alicia Pack, Rustici Software
George Vilches, Rustici Software
Jim Ingram, Rustici Software

Contributors

Andy Johnson, Advanced Distributed Learning (ADL) Initiative
Ben Catchpole, Learning Technologies Group
Michael Martindale, Learning Technologies Group
Jared Orlin, Learning Technologies Group
Stephanie Woor, Learning Technologies Group
Megan Bowe
Tara Morey, Rustici Software
Tim Edwards, Rustici Software

Designated Department of Defense Stakeholder Representatives

Dr. Mitchell L. Bonnett, 83rd United States Army Reserve Readiness Training Command
Shawn Miller, Defense Acquisition University (DAU)
Gary R. (Hank) Reeves, USN PEO EIS
Richard S. Shipmon, Army University

Contents

Introduction to cmi5 and E-learning Standards

cmi5 for Content Professionals

cmi5 for Systems Professionals

Acronyms

ADL	Advanced Distributed Learning
AICC	Aviation Industry Computer-Based Training Committee
API	Application Programming Interface
AU	Assignable Unit
CaaS	Content as a Service
CATAPULT	cmi5 Advanced Testing Application and Player Underpinning Learning Technologies
CLI	Command Line Interface
cmi5	The name of the xAPI Profile
CTS	cmi5 Content Test Suite
DADLAC	Defense Advanced Distributed Learning Advisory Committee
DID	Data Item Description
DL	Distributed Learning
DoD	Department of Defense
DoDI	Department of Defense Instruction
IFI	Inverse Functional Identifier
IRI	Internationalized Resource Identifiers
IRL	Internationalized Resource Locator
JSON	JavaScript Object Notation
LMS	Learning Management System
LTS	cmi5 LMS Test Suite
LRP	Learning Record Provider
LRS	Learning Record Store
PDF	Portable Document Format
PII	Personally Identifiable Information
REST	Representational State Transfer
RFI	Request for Information
RFP	Request for Proposal
SCORM®	Sharable Content Object Reference Model®
TLA	Total Learning Architecture
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
xAPI	Experience Application Programming Interface
XML	Extensible Markup Language

Introduction to cmi5 and E-learning Standards

In this section:

- Modernizing learning ecosystems and the role of cmi5
- The evolution of e-learning standards
- cmi5: Transitioning to a modern ecosystem

Purpose of this section:

This section provides an overview and background of the legacy SCORM standard and the more modern xAPI specification that the DoDI 1322.26 states is the preferred learning and data tracking standard.

This section answers:

- How does cmi5 fit in with modernization efforts?
- What are SCORM and xAPI?
- What is cmi5?
- What are the benefits and use cases for cmi5?

Chapter 1

Modernizing Learning Ecosystems and the Role of cmi5

In this chapter:

- Modernizing learning and training within the Department of Defense (DoD)
- Overview of cmi5 and how it fits into modernization efforts
- About this Best Practices Guide

Modernizing Learning and Training Within the Department of Defense

From augmented reality and virtual reality, to simulations and gamification, advancing technologies are rapidly changing the learning landscape. The wide variety of modalities and ways learners consume training looks far different from the browser-based, text-heavy formats of the past. But these advances have called attention to the limitations of traditional e-learning standards and the SCORM® (Sharable Content Object Reference Model) standard as they are not extensible enough to support emerging technologies and do not provide the portability of data necessary to truly support the goal of having an interconnected learning ecosystem.

Seeing the limitations of SCORM and the need for more modern learning standards, the Advanced Distributed Learning (ADL) Initiative developed the Total Learning Architecture (TLA) research project. The TLA "defines a set of specifications and standards for connecting these various experiences [advanced technologies, instructional methodologies, technology platforms, and activities] to one another and throughout an individual's lifelong continuum of learning."[1] Multiple services and learning opportunities with various modalities and points of delivery can be managed in an integrated, interoperable environment within this modern learning ecosystem.[2]

Chief Master Sgt. Mark Umfleet, 317th Airlift Wing command chief tests HaptX Gloves Development Kit 2 at Dyess Air Force Base. Introducing virtual reality gloves into a training environment allows students to feel each action as it is performed.

- Photo courtesy of the U.S. Air Force by Airman 1st Class Colin Hollowell.

The Department of Defense (DoD) also recognized this need and renewed DoD Instruction 1322.26 ("Distributed Learning") in 2017 to address emerging learning technology concepts and for transitioning from legacy SCORM-based, LMS-centric courseware to a more modern learning ecosystem.

> "To implement DoD policy affecting distributed learning, the DoD Components will adhere to several guidelines. When developing or acquiring distributed learning, they must search for existing distributed learning content that may be reused or repurposed, and should make existing distributed learning assets, content, and other reusable resources visible and accessible to other DoD Components. They should strive to design and develop distributed learning that leverages learning science, technology, specifications, and standards to produce state-of-the-art, affordable, effective, and convenient education and training... As such, they should implement the Experience Application Programming Interface (xAPI) and associated Learning Record Store (LRS) capabilities, as practical, to enhance learning data security and interoperability. Further, the DoD Components are compelled to record, analyze, measure, manage, and, as appropriate, exchange learning experience data among themselves as well as measure and evaluate learner performance."
>
> - Advanced Distributed Learning (ADL) Initiative summary[3] of DoDI 1322.26

The Experience API (xAPI) is one of many interdependent components needed to enable modernization efforts across the DoD and federated platforms, but there's a large divide between legacy tools and xAPI when it comes to migrating existing SCORM content.

Overview of cmi5 and How It Fits Into Modernization Efforts

cmi5 is an xAPI Profile that provides a set of rules for how online courses are imported, launched, and tracked using a Learning Management System (LMS) and xAPI, which is a key piece missing from the xAPI data standard. cmi5 uses a simplified tracking model specifying only the most essential elements for interoperability across most learning instances, including score, status, and time. While cmi5 only explicitly defines the necessities, it's capable of recording and reporting on much more.

To think of cmi5 another way using an analogy:

> "Imagine that xAPI is like the electrical service in your home, and cmi5 is like a wall socket. The wall socket is a simple "plug and play" standard for using electricity with consumer appliances (a very common "use case"). The additional rules imposed by the wall socket for connecting to the electric wiring make it much easier to use. You don't have to have special knowledge or hire an electrician – you just plug it in and it works (as long as the appliance and the wall socket follow those 'extra rules')."
>
> - Bill McDonald, cmi5 Working Group Leader[4]

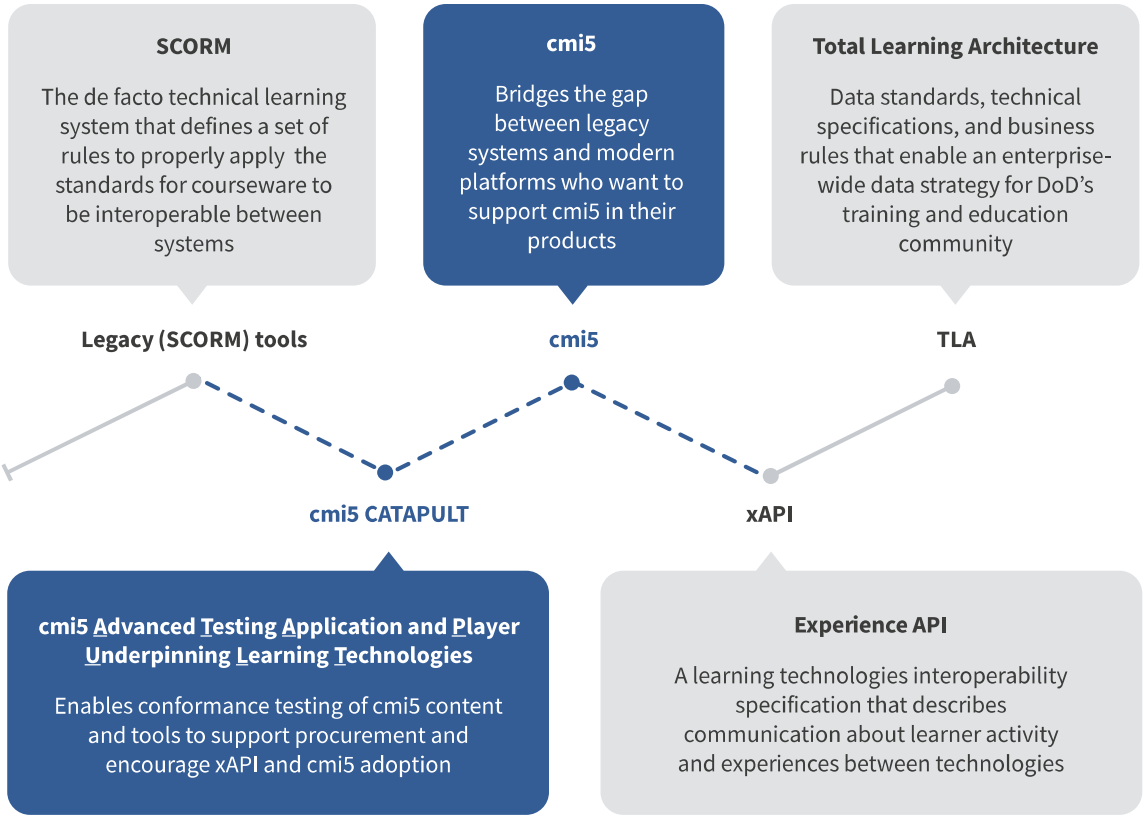
The cmi5 specification was designed to bridge the divide between SCORM and xAPI by reproducing the functionality of SCORM while leveraging the technology benefits that xAPI supplies. The purpose was to replace SCORM as the de facto format for online courses and traditional computer-based training.

> "cmi5 will play an important role in DoD modernization by facilitating progress from SCORM-based learning management system (LMS)-centric courseware to a distributed learning 'ecosystem' that delivers diverse learning opportunities across a range of federated platforms."
>
> - Advanced Distributed Learning Initiative[5]

cmi5 and xAPI will help with the move to a more modern learning and training environment. In 2021, the ADL Initiative released a suite of tools and resources as part of the cmi5 CATAPULT (cmi5 Advanced Testing Application and Player Underpinning Learning Technologies) project to help facilitate the migration of legacy courseware to cmi5 content and accelerate adoption of cmi5 conformant courseware and systems across the DoD, government agencies, and industry. As part of cmi5 CATAPULT, the following resources were developed and are now available:

- cmi5 Content Test Suite (CTS) with user documentation for testing courseware
- cmi5 LMS Test Suite (LTS) with user documentation for testing systems
- Example course templates to help migrate legacy content
- Open-source player prototype with implementation documentation
- cmi5 Best Practices Guide with acquisition and implementation information

Figure 1-1: Where Do cmi5 and the CATAPULT Conformance Tools Fit?



Source: Rustici Software

About This cmi5 Best Practices Guide

Migrating to a new standard requires a lot of thought, consideration, and planning. This guide is intended to help you understand where, when, and how to incorporate cmi5 into your Distributed Learning (DL) strategy. As you navigate the transition, it will also provide guidance to ensure that your cmi5 implementation is successful and helps you move closer to realizing the benefits of broader, more distributed learning and training initiatives.

This guide will explain what cmi5 is, when to use the specification, why it's important for learning and development programs, and the best practices for implementation and acquisition of content and systems. Further, this guide will help program managers, content administrators, instructional designers, content authors, content testers, and developers understand how cmi5 impacts their role as well as answering questions that may arise. The goal of this guide is to help accelerate adoption of the cmi5 specification, which will ultimately support the vision of the TLA and modernizing learning across the DoD.

Using This Guide

This guide serves as a resource for all personnel involved in working on cmi5 related projects including, but not limited to: content migration, content creation, and acquisition and implementation of conformant systems and tools. Contained within this guide are best practices that apply to the cmi5 specification with the purpose of providing a clear and straightforward path to meeting all cmi5 requirements for conformance, content and systems acquisitions, migrating legacy content, and building cmi5 content and systems. This guide also introduces the specific resources that support the implementation of cmi5, including the cmi5 CTS, cmi5 LTS, the cmi5 Player Prototype, and example cmi5 course templates. The following subsections discuss how to read this guide and what to expect.

Organization

This guide is broken down into three main sections with additional Endnotes as follows:

- **Introduction to cmi5 and E-learning Standards:** This section provides an overview and history of SCORM and xAPI standards as well as an introduction to, and basics of, the cmi5 specification.
- **cmi5 for Content Professionals:** This section is for content professionals looking for more in-depth information about the cmi5 specification and best practices for incorporating cmi5 into a content strategy. Topics covered in this section include:
    - Best practices for applying cmi5 within learning activities
    - Acquiring cmi5 content and content creation tools (authoring tools) from vendors
    - Example contracting language for content acquisitions, implementing cmi5 content, and/or building cmi5 content
    - Migrating legacy SCORM content to cmi5 using the course template examples
    - Testing learning activities in the CTS
- **cmi5 for Systems Professionals:** This section is for systems professionals seeking information about acquiring cmi5 conformant systems, example contracting language for systems, the cmi5 Player Prototype, and cmi5 LTS.
- **Endnotes:** Footnote references are made throughout this guide, and the sources and links for the footnotes can be found in the Endnotes section.

Intended Audience

The information in this guide applies primarily to any personnel involved in the creation and/or acquisition of cmi5 content or systems, including, but not limited to, program managers, content administrators, instructional designers, content authors, content testers, content developers, and systems developers.

Required Knowledge

This guide assumes you have some familiarity, working knowledge, or are seeking more information about one or more of the following areas:

- E-learning standards, such as SCORM and xAPI
- General usage of LMSs
- The DoD's TLA mission

While the above list is not exhaustive, it covers the major areas of requisite knowledge for the contents of this guide. There are many available resources to assist in better understanding the above areas. Chapter 2 consists of an overview of SCORM and xAPI, and Chapters 3 and 4 provide an introduction and deep dive into the cmi5 specification.

Resources

The following resources will be helpful as you are learning about the cmi5 specification, acquiring cmi5 content and systems, and the tools available to help with implementation.

cmi5 and xAPI Resources

- **cmi5 specification:** See the full cmi5 specification describing the interoperable runtime communication between the LMS and Assignable Units (AU). https://github.com/AICC/CMI-5_Spec_Current/blob/quartz/cmi5_spec.md
- **The cmi5 Project:** Provides a cmi5 overview as well as information for developers, such as frequently asked questions and design models. https://aicc.github.io/CMI-5_Spec_Current
- **ADL's cmi5 page:** Includes an overview and history of the standard as well as helpful resources. https://adlnet.gov/projects/cmi5-specification/
- **xAPI standard:** View the full xAPI standard. https://github.com/adlnet/xAPI-Spec

DoD and TLA Resources

- **DoDI Instruction 1322.26:** Review the Instruction for Distributed Learning (DL) in its entirety. https://www.esd.whs.mil/Portals/54/Documents/DD/issuances/dodi/132226_dodi_2017.pdf?ver=2017-10-05-073235-400/home_new
- **DoDI Instruction 1322.26 fungible references:** Refer to the official reference and support resource for DoDI 1322.26. https://adlnet.gov/policy/fungible/
- **Total Learning Architecture (TLA):** Discover technical specifications and guidance for integrating learning technologies. https://adlnet.gov/guides/tla/

cmi5 CATAPULT Resources

- **cmi5 CATAPULT:** Contains the latest information about the cmi5 CATAPULT tools and resources available. https://adlnet.gov/projects/cmi5-CATAPULT/
- **cmi5 CATAPULT documentation:** Access all of the user documentation, course templates, and CATAPULT resources. https://adlnet.github.io/CATAPULT

Chapter 2

The Evolution of E-learning Standards

In this chapter:

- The evolution of e-learning standards
- SCORM: The legacy standard
- xAPI: A modern approach
- What's next for e-learning standards?

The Evolution of E-learning Standards

When the learning and training industry transitioned from courses and content that were computer-based, like CDs or floppy disks, to web-based online training, it became clear that there were challenges in terms of playing content and interoperability between systems. As more systems were introduced, organizations found themselves having to rebuild content to adapt courses to the context of each new system, which became both inefficient and expensive. A more portable and scalable solution was needed to support the reusability of content across systems. In 2000, the Advanced Distributed Learning (ADL) Initiative created the SCORM standard to address e-learning interoperability, reusability, and durability challenges.[1]

Today, a similar transition is occurring with emerging technologies, such as simulations, virtual reality, augmented reality, and gaming, which are changing the learning and training landscape. SCORM is not flexible or extensible enough to adequately support a modern and distributed learning ecosystem. In an effort to modernize, the Department of Defense (DoD) updated the DoDI 1322.26 Distributed Learning (DI) in 2017 to incorporate the more modern xAPI standard.

> *"To implement DoD policy affecting Distributed Learning (DL), the DoD Components will, when developing or acquiring DL, implement the Experience Application Programming Interface (xAPI) and associated Learning Record Store capabilities, as practical, to enhance learning data security and interoperability."*
>
> - DoD Instruction 1322.26 Distributed Learning (DL) (2017, p.6)[2]

The ADL Initiative is also working on the Total Learning Architecture (TLA), a research and development project, that includes a "set of technical specifications, standards, and policy guidance that define a uniform approach for integrating current and emerging learning technologies into a learning services ecosystem."[3] The goal is for multiple services and learning opportunities to be managed in an integrated, interoperable "plug and play" environment within that ecosystem.

SCORM: The Legacy Standard

First introduced in 2000, the technical learning standard SCORM has been the de facto standard for both government agencies and industry. At its core, SCORM allows for orderly online training and for content creators to efficiently distribute their courseware to a variety of Learning Management Systems (LMSs). SCORM was intentionally designed to meet the DoD's requirements for creating interoperable, browser-based learning content.[4]

What Is SCORM?

SCORM defines a set of rules for developers on how to properly apply the standards in order for courseware to be interoperable between systems.[5] Meaning, the standard governs how different content, platforms, and systems, such as an authoring tool exporting courseware to an LMS, "talk nicely" to each other and can share and transfer information. SCORM helps LMSs handle and play a variety of different content pieces from various sources.

SCORM

Sharable **C**ontent **O**bject **R**eference **M**odel

A set of technical standards for e-learning products

The SCORM standard specifies a common data model and application program interface (API) for e-learning content. This combination of data model and API allows for standardized communications between client-side content and a system component (called "the run-time environment"), which is commonly provided by an LMS.

SCORM has gone through several iterations since it was created, and three versions are relevant and supported by the ADL Initiative today:

- SCORM 1.2
- SCORM 2004 3rd Edition
- SCORM 2004 4th Edition

Introduced in 2001, SCORM 1.2 is currently the most widely used version and is supported by almost every commercial authoring tool, content provider, and LMS on the market. SCORM 1.2 captures the "big four" data points: completion, score, duration, and satisfaction. SCORM 1.2 also supports the collection of limited interaction data.

The phrase "SCORM 2004" refers to any of the 2004 Editions, but each one differs slightly, with the addition of sequencing and navigation being the most notable enhancements from SCORM 1.2. While four editions were initially introduced, SCORM 2004 3rd and 4th editions are the most prevalent and supported.

Known Implementations of SCORM

As the most widely adopted e-learning standard across government, academia, and industry, government agencies have created or procured vast libraries of SCORM conformant content and employ a variety of SCORM conformant LMSs and training systems to meet the unique needs of each department.

Historically, the DoD and government agencies, such as the U.S. Department of the Navy, have used the SCORM standard when any of the following related to e-learning content are true:

- Delivered only in a web browser on a desktop or laptop computer.
- Makes use of commercial authoring tools that only export to SCORM.
- Custom content with a dependency on SCORM-specific LMS capabilities, such as sequencing and navigation.
- Hosted in the LMS and uses the LMS for reporting.[6]

Limitations of SCORM

SCORM's limitations have become increasingly apparent in the last several years as both technology and internet best practices have evolved. The specification does not provide a data model flexible enough to adequately capture both learning and performance data or how the learner interacts within a course. Additionally, SCORM is limited to only tracking browser-based learning content housed within the LMS. This means that the following are not supported: newer technologies (such as virtual reality), content that isn't browser-based (such as simulations), and mobile activities and applications that reside outside of the LMS or in a distributed fashion. In addition, SCORM content must reside on the same domain as the LMS. SCORM is also very limited in what it can natively track. Finally, as web languages and best practices have evolved, the standard hasn't been flexible enough to stay current with these changes.[7]

In summary, the limitations of SCORM include:

- ✗ Only operates within a browser context.
- ✗ Cannot track "anything" and is constrained to a defined set of data elements.
- ✗ Cannot track mobile applications natively.
- ✗ Content must reside on the same server or domain as the LMS.
- ✗ Data must be collected and stored within the LMS.

While SCORM is currently the most widely used learning specification, the xAPI standard and cmi5 specification have been introduced because they are flexible enough to provide robust data tracking and are extensible enough to work with emerging technologies and the ever-growing distributed learning landscape.

xAPI: A Modern Approach

While SCORM has been the de facto standard, the DoD and industry are moving to a more modern learning environment with diverse tools and technologies. In 2013, the xAPI standard was released to meet the changing needs of advancing technologies and to provide a way to track a variety of formal and informal learning experiences in and beyond computer-based training.

xAPI

Experience **A**pplication **P**rogramming **I**nterface

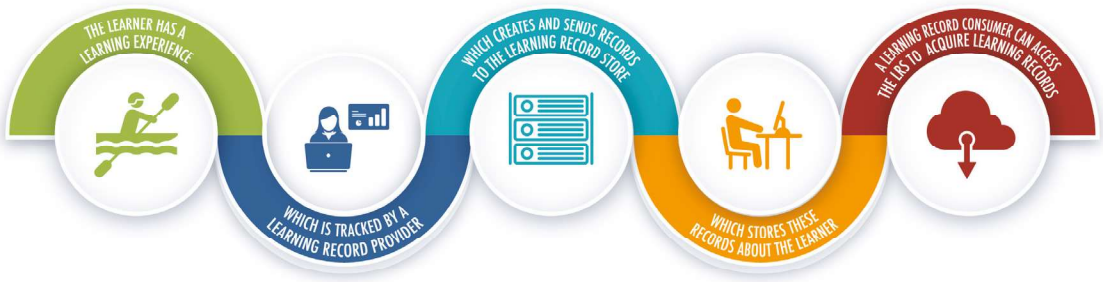
Data and interface standard for capturing data.

What Is xAPI?

xAPI is an open-source data and interface standard that specifies a structure describing learning experiences and defines how these learning experience descriptions can be captured and shared electronically between different systems.[8] In other words, xAPI allows for capturing much more expressive data about virtually any activity or learner experience and has the ability to share that data across systems.

One of the primary goals of xAPI is to make analyzing and comparing learning experiences and outcomes as easy as possible, even when activities are recorded in disparate contexts, systems, or technologies. The standard also strives to maximize interoperability between platforms and services that produce, collect, store, and analyze learning data. These goals are especially key for government agencies or organizations needing to share content and data with other departments that may be using different platforms or systems.

Figure 2-1: Tracking Experiences with xAPI



Source: The Experience API Specification, 2013, Sect. 5.0 xAPI Components[9]

Key Components of xAPI

xAPI provides a blueprint for developers building applications that need to implement or conform to the specification. As a technical specification, xAPI aims to facilitate the documentation and communication of learning experiences by specifying a structure to describe learning experiences and defining how these descriptions can be exchanged electronically. To make this happen, there are two key components of the xAPI standard: xAPI Statements representing learning data and an LRS (Learning Record Store) for storing and sharing Statements.

xAPI Statements

xAPI Statements, or learning records, can serve the role of providing evidence for the experiences and events tracked using xAPI. Statements follow a machine-readable JSON (JavaScript Object Notation) format, but they can also be described using natural language, which is handy for the design process. The larger purpose of the xAPI Statement is to provide meaning for the overall learning experience, not just the sum of its parts. Statements must include the following:

An actor (the learner)

A verb (the action)

An object (the activity)

But data tracking doesn't have to stop there. xAPI Statements can be expanded to include even more details, including context (like who the instructor was and the location of the learner during the activity), what happened within the activity or course (such as showing where a learner spent extra time on a module), and the result or score.

Here are two examples to show the range of data that can be collected:

Required: Sally (**actor**) experienced (**verb**) Solo Hang Gliding (**activity**).

```
{
  “actor”: {
    “name”: “Sally Glider”,
    “mbox”: “mailto:sally@example.com”
  },
  “verb”: {
    “id”: “http://adlnet.gov/expapi/verbs/experienced”,
    “display”: { “en-US”: “experienced” }
  },
  “object”: {
    “id”: “http://example.com/activities/solo-hang-gliding”,
    “definition”: {
      “name”: { “en-US”: “Solo Hang Gliding” }
    }
  }
}
```

Expanded: Sally completed Solo Hang Gliding with a score of 95% (**result**).

```
{
  “actor”: {
    “name”: “Sally Glider”,
    “mbox”: “mailto:sally@example.com”
  },
  “verb”: {
    “id”: “http://adlnet.gov/expapi/verbs/completed”,
    “display”: { “en-US”: “completed” }
  },
  “object”: {
    “id”: “http://example.com/activities/solo-hang-gliding”,
    “definition”: {
      “name”: { “en-US”: “Solo Hang Gliding” }
    }
  },
  “result”: {
    “completion”: true,
    “success”: true,
    “score”: {
      “scaled”: .95
    }
  }
}
```

(The code above is for example purposes only and not intended for use.)

Learning Record Store (LRS)

The LRS plays a vital role in the digital learning ecosystem since it receives, stores, and provides access to the learning records (xAPI Statements). An LRS is necessary when using xAPI and can be used as a stand-alone service or be embedded in an LMS or other system to collect the xAPI Statements. LRSs also validate the format of the Statements to ensure only Statements that are conformant to the standard are accepted and retained.

Figure 2-2: Stand-alone LRS

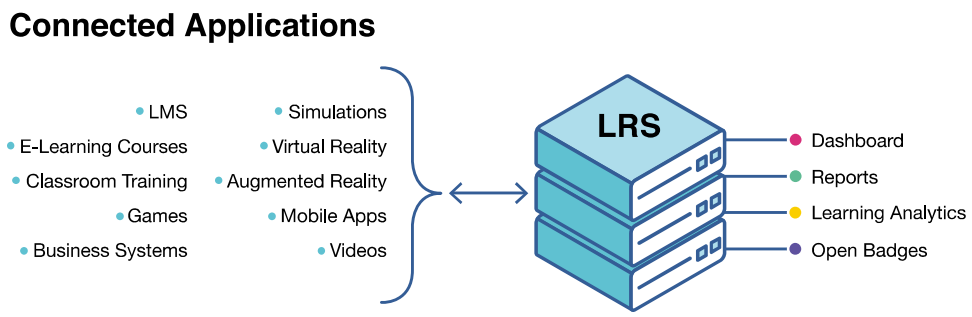
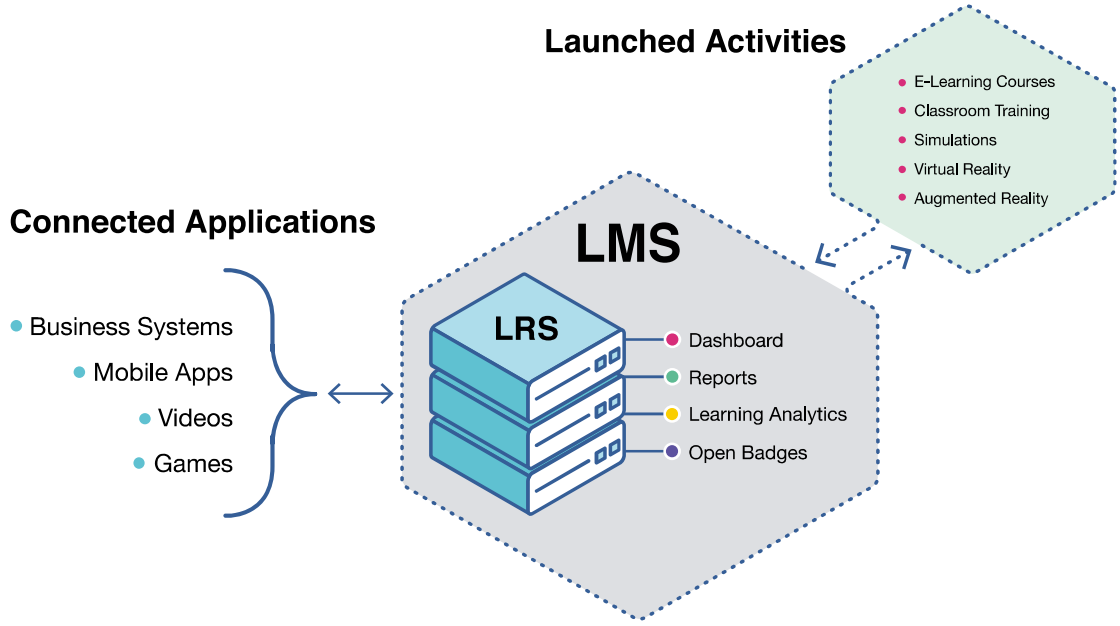


Figure 2-3: An LRS in an LMS



Source: Rustici Software

Known Implementations of xAPI

The Department of Defense (DoD) Instruction 1322.26 declares that the xAPI data standard is the superior strategy for tracking learning data. To move toward the envisioned lifelong continuum of learning for service members and DoD workers, it's vital to have a learning standard that can accurately and adequately capture learner data across systems and experiences.

Examples of when to use xAPI:

- When an LMS isn't required.
- When the content, course, or training is not browser-based.
- When tracking simulations, virtual reality, games, or mobile activities.
- When tracking specific data beyond SCORM's big four is preferable or necessary.
- When the content creator wants to maintain control of the content.

If any of the following contracting requirements for e-learning content are true, then U.S. Department of the Navy lists the following as distinguishing factors for when xAPI is required:

- Delivered on any device (e.g., desktop, mobile).
- Custom content that implements the NETC xAPI Profiles.
- No dependencies on SCORM features in an LMS.
- Hosted anywhere (e.g., distributed content server, LMS, CMS, private network).
- Accessed from anywhere, not just an LMS (e.g., personal device, bring your own device, disconnected/offline).
- Must have better resolution into the data and associated analytics.[10]

Instances of xAPI from the Defense Advanced Distributed Learning Advisory Committee (DADLAC) 2020 Annual Report:[11]

(Photo courtesy of the U.S. Army, 2019, Augmented Reality Training)

The Army Distributed Learning Program

"ADLP has developed innovative and engaging courseware products using the latest instructional technologies. Immersive products being developed include 3D models delivered in virtual environments, virtual and augmented reality, xAPI with instructional dashboards, level 3 development of foreign language courses, stealth assessments, and gamification."

- Helen Remily, Army Distributed Learning Program Director

Defense Acquisition University

"Recently, DAU has focused on modernizing its capabilities including research into xAPI Learning Record Store design and use cases, xAPI applications for personalization and data analytics pilot projects, and AI and adaptive learning capabilities."

- Shawn Miller, Technology Innovation Transformation Program Manager

Marine Corps Training and Education Command

"Currently, TECOM is working on several efforts to improve its learning architecture. One effort is an evaluation of xAPI Learning Record Stores, with the goal of identifying the education and training data needed to get a better snapshot of a Marine's overall training."

- Larry Smith, Marine Corps University Director of Ed Tech/College of Distance Education and Training Technical Director

(Photo courtesy of the U.S. Marine Corps by Sgt. Jesus Sepulveda Torres, 2021, Convoy Simulator)

Limitations of xAPI

One of xAPI's biggest limitations is that, on its own, the xAPI specification does not define how online courses are structured or how the learning content communicates with the system that is hosting that content. xAPI also does not specify authentication protocols for how the learner connects with content or a course. These definitions and protocols are vital for launching and tracking courses using an LMS, and the lack of instructions has slowed adoption rates.

Another limitation is that while xAPI Statements can capture a virtually limitless amount of data, it becomes challenging for systems to extrapolate and analyze that data in a meaningful way without a defined vocabulary and instructions. For instance, the verb "performed" can have several different meanings depending on the context:

> Sally **performed** CPR.
>
> Sally **performed** a PIT maneuver.
>
> Sally **performed** with the Air Force Band.

In summary, the limitations of xAPI include:

- ✗ Doesn't include instructions for launching content in an LMS.
- ✗ Doesn't have established learning activity rules for normalized reporting.
- ✗ Doesn't have defined completion criteria.

xAPI Profiles

Without specific definitions, these Statements could be recorded and analyzed incorrectly. The xAPI Profile Specification was introduced to overcome this risk by defining vocabulary, contexts, and rules for how xAPI data interacts within a particular domain.[12] This data can then be used to help learning and training professionals better understand the behaviors of their learners.

Essentially, xAPI Profiles are roadmaps that drive successful implementation and semantic interoperability of xAPI data across systems. However, in order to have wider interoperability across platforms and systems, a common vocabulary or profile is needed for larger audiences to better understand the data in an xAPI Statement.[13]

xAPI Profile Specification

Establishes rules for individual xAPI Profiles to leverage concepts, extensions, statement templates, and statement patterns to present clear and consistent data that's readable by both humans and technologies.

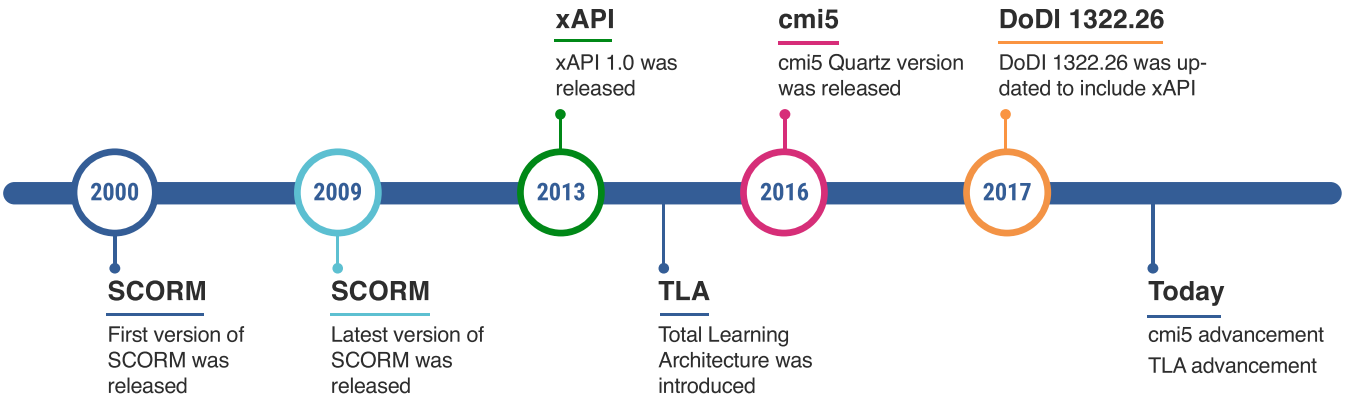
Example of a Video Played Statement With All the Required and Recommended xAPI Profile Elements Highlighted

```
{
  "actor": ...,
  "verb": {
    "id": "https://w3id.org/xapi/video/verbs/played", ...
  },
  "object": {
    "definition": {
      "type": "https://w3id.org/xapi/video/activity-type/video"
    }, ...
  },
  "result": {
    "extensions": {
      "https://w3id.org/xapi/video/extensions/time": 0
    }
  },
  "context": {
    "contextActivities": {
      "category": [
        {
          "id": "https://w3id.org/xapi/video"
        }
      ]
    }
  },
  "id": "ac522d8a-ed56-4a5e-bc86-14067aec4101"
}
```

(The code above is for example purposes only and not intended for use.)

What's Next for E-learning Standards?

Figure 2-4: Evolution of E-learning Standards and Projects



Source: Rustici Software

Advancing technologies are revolutionizing training and changing what the future of learning looks like. However, traditional LMSs and legacy systems continue to be key components of formal and educational training. In order to embrace these new technologies and training modalities, the DoD has declared the xAPI standard is the preferred method of tracking. However, in order for xAPI-based training to be used within the context of an LMS, a defined set of rules is needed.

In 2016, cmi5 was modified to bridge the gap between legacy SCORM courseware and modern platforms using xAPI. cmi5 allows for more flexibility than SCORM provides while including a defined set of vocabulary and launch instructions that are necessary with LMS use and that are not specifically expressed with the xAPI standard. cmi5 and xAPI are among the many interdependent components needed to enable modernization efforts across the DoD and federated platforms.

Chapter 3

cmi5: Transitioning to a Modern Learning Ecosystem

In this chapter:

- What is cmi5?
- How cmi5 works
- Why cmi5?
- When to use cmi5
- Moving forward

What Is cmi5?

Modernizing learning and training requires newer, more flexible standards than the legacy e-learning SCORM (Sharable Content Object Reference Model) standards can provide. xAPI (Experience API) was introduced to enable a more modern approach to learning and training technologies as noted in the Department of Defense (DoD) Instruction 1322.26 that states that xAPI is the superior tracking strategy for learning data.[1] Learning Management Systems (LMSs) continue to be a key component of learning ecosystems, but xAPI doesn't specify how to handle key components of the LMS architecture. Without these defined instructions, xAPI cannot simply replace SCORM. A more defined set of rules is needed to support xAPI in the context of legacy, SCORM-based systems.

cmi5 is an xAPI Profile that bridges the divide between SCORM and xAPI by providing the necessary requirements, or rules, that inform how an LMS and xAPI-enabled learning activities communicate with each other. The cmi5 specification will play a key role in moving toward a more modern learning ecosystem with plug and play interoperability.

Predating xAPI, cmi5 originally started with the Aviation Industry Computer-Based Training (AICC) in 2010, and it was expected to replace both AICC and SCORM standards. cmi5 was "rebooted" in 2012, and the AICC replaced the previous architecture with xAPI after seeing the benefits and a clearer adoption path. In 2014, the AICC dissolved and formally transitioned cmi5 to ADL where it's still guided by its original goals.

cmi5

(the name of the xAPI Profile)

E-learning specification employing xAPI as a communications protocol and data model while providing definition for components necessary for use with LMS

cmi5 was once again modified in 2016. This time with the intent to enable the transition from SCORM-based content and systems while leveraging xAPI by defining how to use xAPI in the context of a traditional launching system or LMS. The cmi5 specification defines how learning and training resources are imported, launched, and tracked in a way that is similar to SCORM while providing more advanced opportunities by conforming to the xAPI specification. Specifically, cmi5 uses xAPI as the communication and data layer and also implements controlled vocabularies, which are necessary for plug and play interoperability between LMSs and LMS-like systems. Essentially, cmi5 is an xAPI Profile that contains a vocabulary model and xAPI Statement patterns.[2]

cmi5 was specifically designed for the LMS use case where the learner launches the learning content or activity from an LMS user interface. cmi5 specifies interoperability rules[3] for handling:

- Content launch mechanism
- Authorization
- Session management
- Reporting
- Course structure

These defined rules ensure interoperability between disparate learning activities and systems that xAPI doesn't specify. Like xAPI, cmi5 has the ability to record activities performed outside of an LMS, but unlike xAPI, those activities must be launched by an LMS with cmi5.

cmi5 = xAPI + LMS

cmi5 incorporates a simplified tracking data model by only specifying the essential elements needed to be interoperable across most learning instances, such as score, status, and time. While cmi5 only explicitly defines the necessities, it's capable of recording and reporting or retrieving content-defined data, which permits content authors and designers to add features in the future without sacrificing plug and play interoperability.[4] Unlike SCORM, cmi5 is not dependent on a browser to communicate or launch courses, content, or experiences.

> *"cmi5 is a floor rather than a ceiling. It establishes the baseline, and you can make any other statement you want. The power of xAPI with interoperability."*[5]
>
> - Bill McDonald, cmi5 Working Group Leader

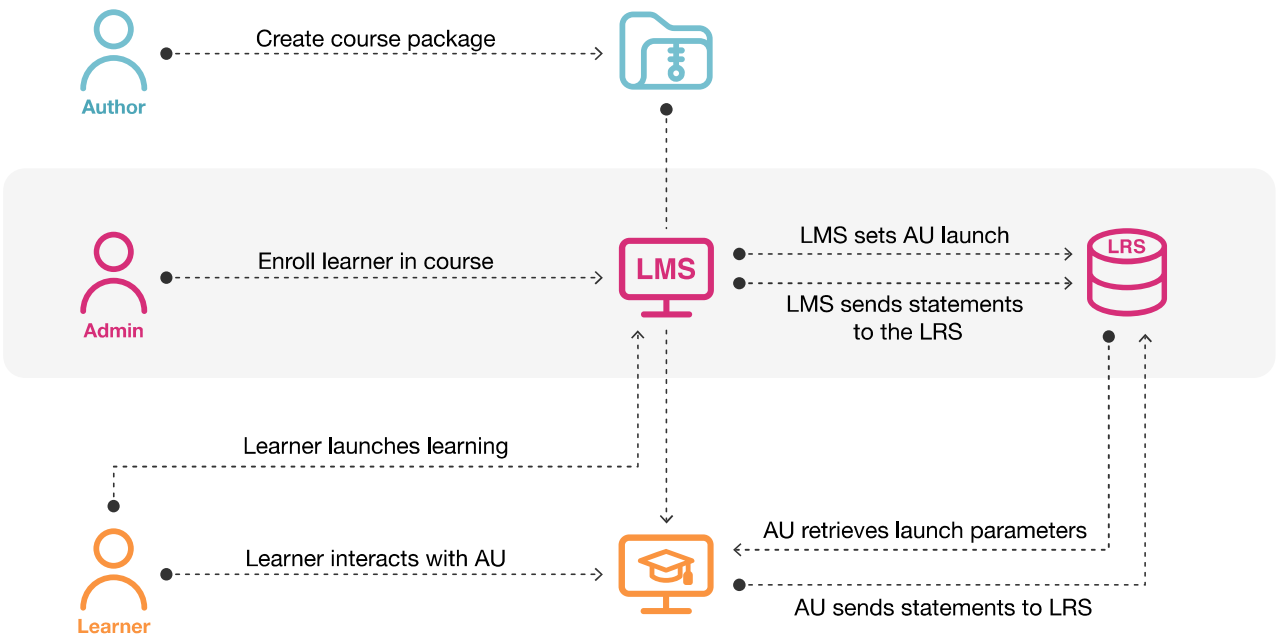
How cmi5 Works

There are four elements that play key roles in cmi5:

- Assignable Unit (AU)
- Course Package
- Learning Management System (LMS)
- Learning Record Store (LRS)

cmi5 utilizes xAPI as the communication and data layer to track and send data about learning activities to an LRS.

Figure 3-1. Illustration of How cmi5 Works



Source: The cmi5 Project[6]

Assignable Units (AU)

Each learning activity within the cmi5 course structure is called an Assignable Unit (AU). AUs are the launchable pieces of content that include the concepts of completion, pass/fail, score, and duration. For example, the AU may include a moveOn value to determine if the AU was successfully completed. Each course requires at least one AU.

The AU's responsibilities are as follows:

- Parse the parameters from the launching environment to determine where the LMS location is and initiate communication with the LMS.
- Act as a "client," sending and receiving messages using the defined transport mechanism(s) and associated commands as prescribed in this specification.
- Format all data according to the defined data types and vocabularies that are defined in this specification.
- Send a "Terminate" statement prior to terminating the AU's execution.[7]

Further, AUs are required to do the following per the cmi5 specification:

- Implement runtime communication as defined in the xAPI Specification.
- Implement Statement API requirements as defined in the AU Statement API requirements listed in Section 7.1.
- Implement the runtime launch interface as defined in the content launch mechanisms in Section 8.
- Implement State API requirements in this specification as defined in the xAPI state data model in Section 10.
- Implement Profile API requirements in this specification as defined in the xAPI agent profile data model in Section 11.[8]

Course Package

The course package is essentially a collection of all of the AUs that make up a course and their launch parameters. It represents the course structure and acts as a directory of sorts or a package of references to each component or AU. A course package is an XML (Extensible Markup Language) file and must be called `cmi5.xml` when saved in a ZIP file.[9]

One of the goals for cmi5 is to provide the ability to "package" a course and import it into an LMS. This is similar to SCORM but has a few distinct differences including:

- LMSs that are conformant to the cmi5 specification must be able to import a course with at least 1,000 AUs.
- The course package can reference content or media that's stored remotely. While the course structure is imported into the LMS, the content or resources may reside in other domains.
- Satisfaction criteria, or "moveOn" criteria, are more clearly defined than SCORM.[10]

Learning Management System (LMS)

With cmi5, learners use a Learning Management System (LMS) to launch cmi5 content, and the LMS registers learners, launches the content, tracks the learner's progress, and analyzes and reports on learner performance. cmi5 is specifically designed to address launching a course from the context of an LMS and works with an LRS to track learner data throughout the cmi5 session.

Learning Record Store (LRS)

Learning Record Stores (LRSs) are a key part of cmi5 because they receive, store, and retrieve the learning records (xAPI Statements) sent by the LMS and the AU.[11] LRSs may be stand-alone services or they may be embedded within the same LMS that is launching the course and tracking learning experiences. LRSs also validate the Statements to be sure the format of the Statements are conformant to the xAPI standard. LRSs do not accept or retain Statements that do not conform to the xAPI specification.

xAPI Statements

cmi5 uses xAPI Statements, or learning records, to dictate the format for the specific moments in a learning activity[12] like xAPI does. xAPI Statements convey an experience which has occurred, or may be occurring, in a natural language that is easy for both humans and machines to read. Statements follow a machine-readable JSON (JavaScript Object Notation) format, but they can also be described using natural language, which is handy for the design process. The larger purpose of the xAPI Statement is to provide meaning for the overall learning experience, not just the sum of its parts. Statements must include the following:

An actor (the learner)

A verb (the action)

An object (the activity)

cmi5 statements fall into two categories:

“cmi5 defined” statements

xAPI Statements that are highly defined by the specification and are tied to the concept of session/registration management. They include a specific cmi5 category Activity for easy detection and access in the LRS activity stream.

“cmi5 allowed” statements

xAPI Statements that include the session ID and potentially additional information included in the launch data but are otherwise custom to an AU.[13]

Example of a cmi5 Statement with Required Information Highlighted:

```
{
  “id”: “b476834b-45ea-4f64-a12e-8cc501ab3f55”,
  “actor”: ...,
  “timestamp”: “2021-07-08T20:43:07.272Z”,
  “context”: {
    “contextActivities”: {
      “grouping”: [
        {
“id”: “https://w3id.org/xapi/cmi5/course/geology-intro-single-au-frame/1”
        }
      ],
      “category”: [
        {
          “id”: “https://w3id.org/xapi/cmi5/context/categories/cmi5”
        },
        {
          “id”: “https://w3id.org/xapi/cmi5/context/categories/moveon”
        }
      ]
    },
    “extensions”: {
      “https://w3id.org/xapi/cmi5/context/extensions/sessionid”:
“dcb0a605-9653-482f-a 8f-56e111441ddb”
    },
    “registration”: “cf046f00-ba66-4c7e-8fa0-03543a5a59eb”
  },
  “object”: {
    “objectType”: “Activity”,
    “id”:
“https://w3id.org/xapi/cmi5/course/356f8ce8-4a12-47ab-95ed-e9ac7f501693/au/0”
  },
  “verb”: {
    “id”: “http://adlnet.gov/expapi/verbs/passed”,
    “display”: {
      “en-US”:”passed”
    }
  },
  “result”: {
    “score”: {
      “scaled”: 1,
      “raw”: 100,
      “min”: 0,
      “max”: 100
    },
    “success”: true,
    “duration”: “PT9S”
  }
}
```

(The code above is for example purposes only and not intended for use.)

cmi5 Verbs

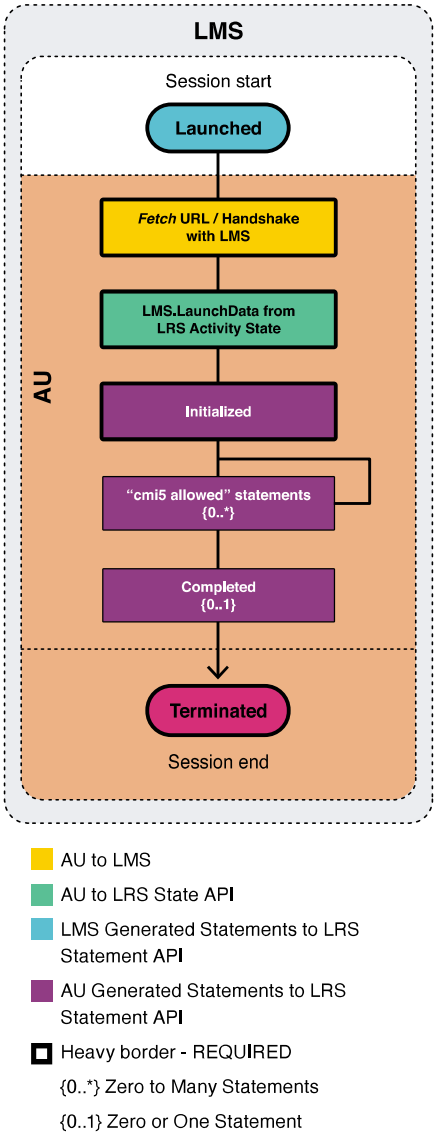
The cmi5 specification defines nine Verbs for "cmi5 defined" statements, but AUs may use additional Verbs that aren't listed. Regardless of the Verbs used, the LMS must record and report all "cmi5 defined" and "cmi5 allowed" statements that aren't rejected.

The nine "cmi5 defined" Verbs are:[14]

Figure 3-2: Nine cmi5 Verbs

Launched	AU was launched by the LMS.
Initialized	cmi5 session fully initialized.
Completed	Learner did all relevant AU activities. Indicates progress of 100%.
Passed	Learner succeeded in a judged AU activity.
Failed	Learner failed in a judged AU activity.
Abandoned	AU session was abnormally terminated.
Waived	LMS determined AU requirements were met by other means.
Terminated	Learner terminated the AU. No more statements will be sent for the launch session.
Satisfied	LMS determined the Learner met the moveOn criteria for all course AUs.

Figure 3-3: Example of a cmi5 Session Using the "Completed" Verb

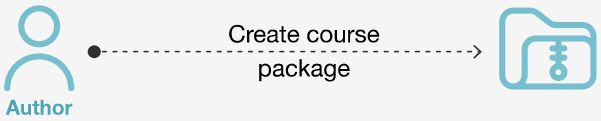


Source: xAPI.com, Rustici Software[15]

cmi5 in Action

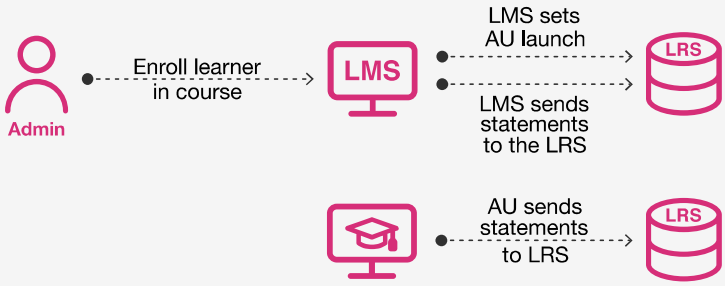
From course creation to reporting, this is the overall process for a typical cmi5 case. The author creates one or more separately launchable AUs, or content, that can communicate using cmi5 and uses xAPI to capture learner data and creates a course structure with instructions for launch, description, and identification. The author also provides completion rules, such as setting a “masteryScore”. The AU’s resources may be referenced via an XML file from an external host or saved as a ZIP file along with the course structure.

Figure 3-4: Author Creates a Course Package



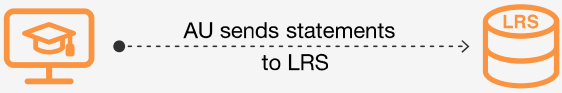
Once complete, the administrator imports the course package into the LMS. The administrator enrolls learner(s) in the course, and the LMS records the enrollment as a registration. The learner initiates the launch of an AU, and the LMS prepares launch data and records the launch in the LRS. The LMS then redirects the learner to the AU presentation, and the learner interacts with the AU.

Figure 3-5: Administrator Enrolls a Learner and LRS Records Registration and Learner Interacts with AU



The AU records activities, scores, mastery, and completion to the LRS.

Figure 3-6: AU Sends Statements to LRS



Why cmi5?

Advanced technologies are impacting training and changing the future of learning. The DoD, government agencies, and organizations want to track and evaluate the impact those trainings have on an employee throughout their lifelong learning journey. Meanwhile, traditional LMSs continue to be a key part of most formal and educational training settings. However, neither SCORM nor xAPI (on its own) can adequately play and track content across disparate technologies and LMSs.

cmi5 opens up the best of both worlds: the flexibility of xAPI for tracking virtually any type of learning activity and the structure of SCORM that learning technologies and systems have historically relied upon. cmi5 will help the DoD, government agencies, and industry vendors transition from legacy models to the future vision of the Total Learning Architecture (TLA) and a more distributed and interoperable learning ecosystem.

Bridging the E-learning Standards Gap

cmi5 bridges the gap between legacy SCORM courseware and modern platforms using xAPI by allowing for more flexibility than SCORM provides while including a defined set of vocabulary and launch instructions that are necessary for use with an LMS.

The xAPI specification does not define course structure or how the learning content communicates with the system that is launching that content. xAPI also does not specify authentication protocols for how the learner connects with a learning activity or course. cmi5 solves these by specifically outlining rules and definitions for course structure, content launch, authorization, session management, and reporting.[16]

cmi5 Benefits

The cmi5 specification's primary focus is on being more extensible, robust, and adaptable to today's technologies than existing standards.[17] Organizations implementing the specification will have the enhanced benefits that xAPI provides while achieving several tracking and interoperability goals.

In addition to the extensibility of being able to track modern technologies, the flexibility to record nearly limitless data, and the interoperability for use across systems and platforms, cmi5 has a few more noteworthy advantages:

- **Interoperability:** Using mechanisms other than cmi5, such as the Launch Profile or SCORM Profile, could cause problems with data portability and normalization if activities are used across federated platforms.
- **Distributed content:** cmi5 supports Content as a Service (CaaS) models of delivery so that content can be stored outside of an LMS.
- **Device/OS/Browser independence:** cmi5 allows for content to reside anywhere, even outside of a browser, to support offline and mobile applications use cases.
- **Data sharing:** cmi5 supports fetching data out of the LRS and displaying the data in the course, if for example, your course requires earlier data to be displayed later.
- **Pop up windows:** No more pop up windows. Content can be launched in the same window as the LMS.[18]

When to Use cmi5

Now that you know a little more about what cmi5 is, how it works, and the benefits it offers, this section will help guide you through when to use the specification.

Comparing the E-learning Standards

Learning and development professionals will want to use the right standard for the right learning activity, but between the versions of SCORM, xAPI, and cmi5, there are a variety of options that could be used. The following table lists a few key features that professionals creating or acquiring training content should be aware of and which standards support those features to help determine whether SCORM, xAPI,[19] or cmi5 is best suited for their needs.

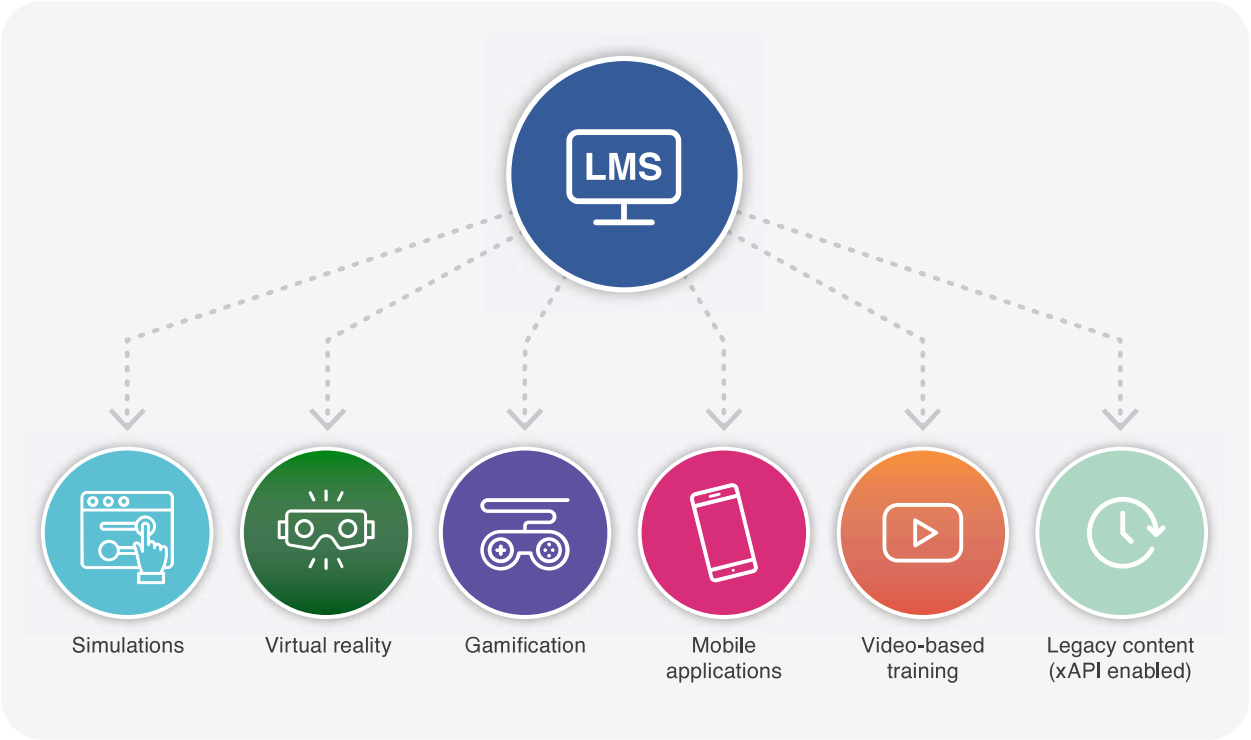
Figure 3-7: Comparison of Key SCORM, xAPI, and cmi5 Features

Features	SCORM	xAPI	cmi5	Description
Defined content launch	✔	✘	✔	SCORM and cmi5 define content launch. xAPI does not include instructions for launching content in an LMS.
Track "anything"	✘	✔	✔	SCORM is constrained to a defined set of data elements while xAPI and cmi5 allow for developer-defined data elements.
Normalized reporting	✔	✘	✔	cmi5 establishes rules for records (statements) to include identifiers for a learner session so that records can be more easily grouped for normalized reports.
Mobile applications	✘	✔	✔	SCORM cannot track mobile applications natively and only operates within a browser context. With cmi5 or xAPI, mobile access to an LMS can be through a mobile web browser directly using the LMS, mobile applications embedding a web browser, or mobile applications using native UI components.
Distributed content	✘	✔*	✔	All SCORM content must reside on the same server or domain as the LMS. Content can be located anywhere with cmi5. *xAPI does not define a content packaging standard, but it does support the concept of a learning activity residing anywhere. SCORM data is collected and stored within the LMS. xAPI and, by extension cmi5, allow for data to be shared across systems.
Data portability	✘	✔	✔	SCORM data is collected and stored within the LMS. xAPI and, by extension cmi5, allow for data to be shared across systems.
Extensibility	✘	✔	✔	With cmi5 and xAPI, any learning data content can be captured. The LMS uses an LRS to support cmi5 and xAPI. With an LRS, you can build a learning ecosystem beyond the LMS, easily connecting to other systems.
Normalized completion criteria	✔	✘	✔	cmi5 establishes interoperable rules for determining completion/ mastery of learning activities. xAPI has no defined completion criteria.
Multiple lesson support	✔	✘	✔	cmi5 packages allow for multiple AUs in a defined hierarchy with criteria for progression, much as SCORM does with multiple SCOs.

Use Cases for cmi5

cmi5 is specifically designed to solve the LMS use case where the learner needs to launch xAPI-enabled content from an LMS-based system.

Figure 3-8: Potential Examples of When to Use cmi5:



In addition, cmi5 can be implemented with all of the same use cases of xAPI with the key distinction that the learner initiates the activity from an LMS.

To help with migrating legacy SCORM content to the cmi5 specification, course templates were developed and are publicly available. These course templates include information for a simple migration of completely browser-based SCORM content to more complex blended courses with a range of instructional activities. The templates focus on the methodology necessary for creating cmi5 courseware. The templates also include instructions for crafting cmi5 course content with the use of open-source libraries without an authoring tool. Once the course has been created, it should be tested in the open-source cmi5 Content Test Suite (CTS) to validate that the content conforms to the specification.

Before procuring content from vendors, ask that the courseware is conformant to the cmi5 specification and that it be successfully validated by the cmi5 CTS. The following chapters dive deeper into acquisition requirements.

Moving Forward

The future of learning and training is evolving. The legacy SCORM standard ensured plug and play interoperability of browser-based courses, but it isn't extensible enough to track new and emerging technologies. cmi5 and xAPI are two components of a modernized learning approach supporting a lifelong learning journey model and addressing the need for a standardized way to share data across systems. The cmi5 specification will play a fundamental role in ensuring that the vision of the TLA and DoDI 1322.26 become realized.

cmi5 will play a key role in the transition from legacy SCORM-based models to modern, xAPI-enabled ecosystems. To facilitate this migration, tools and resources are available to help accelerate the development and validation of cmi5 courseware and systems for conformance.

The following sections in this guide are divided into cmi5 for content professionals and cmi5 for systems professionals.

cmi5 for Content Professionals:

- cmi5 deep dive with information about the cmi5 specification
- Acquiring cmi5 content, including when to buy cmi5 content and best practices
- cmi5 content acquisition guidance with example contracting language
- Implementing cmi5 content with when to use cmi5
- Best practices for building cmi5 content, including using the course templates to migrate legacy content and testing in the cmi5 CTS

cmi5 for Systems Professionals:

- Acquiring cmi5 conformant systems with best practices and what to ask your LMS vendor
- cmi5 conformant systems acquisition guidance and example contracting language
- Information about using and implementing the cmi5 Player Prototype
- Information about using and deploying the cmi5 LMS Test Suite (LTS)

Chapter 4

cmi5 Implementation Guidance

In this chapter:

- How cmi5 fits into DoD modernization efforts
- When to implement cmi5
- Adopting and implementing cmi5
- Using this guide to start implementing cmi5

How cmi5 Fits Into DoD Modernization Efforts

This chapter provides both content and systems professionals with information on how integrating cmi5 content and conformant systems fits into the Department of Defense Instruction (DoDI) 1322.26 to support the vision of the Total Learning Architecture (TLA), when to implement cmi5, and what is needed to implement the specification.

The learning and training industry is changing with the integration of more advanced technologies, and the DoD is modernizing to meet these changes by stating that xAPI is the preferred data tracking standard. However, there's a large divide when moving from legacy SCORM standards to xAPI.

> *"cmi5 applies many of SCORM's familiar rules for interoperability, authentication, content launch, reporting, and other elements, while also enabling the functionality of xAPI to allow broader metadata collection and analytics. The availability of more detailed information on learning activities, including individual and group performance, engagement, and other data points, will enable data-driven decisions for improved education and training, which will save time and money while increasing force readiness."*
>
> - Advanced Distributed Learning (ADL) Initiative.[1]

The DoD has tasked the ADL Initiative and Defense Advanced Distributed Learning Advisory Committee (DADLAC) with the responsibility of routinely maintaining and updating the DoDI 1322.26 fungible references to support the inclusion of xAPI, cmi5, and the TLA vision. Option 2 for transitioning to the xAPI standard lists acquiring and maintaining a cmi5 conformant LMS and an xAPI LRS as one of the DoD approved paths to moving to a more modern learning ecosystem.[2]

When to Implement cmi5

The following subsections list some of the common use cases and opportunities for transitioning to cmi5.

cmi5 Content

Adding simulations and other modern training technologies.
Simulations, virtual reality, augmented reality, and non-browser based applications can significantly enhance learning and training programs, but legacy SCORM standards are not extensible enough to track these newer training modalities. If you want to incorporate these newer modalities into training programs and are launching from an LMS, then cmi5 is the preferred specification for these types of learning activities.

Updating existing content or courseware.
Whenever you need to update or revise existing SCORM or legacy content, convert that existing courseware to cmi5 if your authoring tool is able to publish as cmi5. Doing so will help move modernization efforts forward now and lessen the amount of content that needs transitioning to more modern standards in the future.

Capturing more detailed learning data.
cmi5 is the preferred standard when program managers, content authors, or instructional designers desire to see more detailed data about what happens inside the course or learning activities beyond what the SCORM standards support. cmi5 ensures compatibility and semantic interoperability across systems.

Incorporating xAPI enabled activities.
Transitioning to xAPI offers a lot of data tracking and portability benefits. cmi5 is the preferred xAPI Profile and the perfect vehicle for incorporating xAPI enabled activities that will be launched from an LMS. Using an xAPI Profile other than cmi5, such as the Launch Profile or SCORM Profile may not be sufficient and could cause interoperability problems in the future.

Leveraging authoring tools that support cmi5.
If your current authoring tool provider has cmi5 support, instructional designers and content creators can begin to understand and use the specification in their content now. If the tool doesn't currently support cmi5, program managers and content professionals can start conversations with vendors about adding support.

Ending pop up windows in courseware.
Pop up windows can present multiple issues for learners accessing courseware in their web browser or may prevent access altogether. With cmi5, courses can be launched in the same window as the LMS, eliminating the need for pop up windows.

Acquiring learning activities that are hosted outside of an LMS.
Unlike SCORM, cmi5 supports Content as a Service (CaaS) models of delivery so that content can be stored outside of an LMS. Meaning, as long as the content is launched from the LMS, the content can reside anywhere.

Sharing data across reporting systems.
One of the key benefits of cmi5 is ensuring that data is interoperable across systems. Using an xAPI Profile other than cmi5 could cause interoperability and data portability issues.

cmi5 Systems

Following Option 2 in the DoDI 1322.26 fungible references.[3]
Federated platforms wanting to use an LMS and add xAPI data tracking implement cmi5 content and conformant systems to accomplish these goals.

When the training department or content team wants to begin implementing any of the above instances.
In order to play and track cmi5 learning activities (content), you must have a cmi5 conformant LMS with an LRS. If the training department wants to acquire and implement cmi5, then it is time to procure a cmi5 conformant system or talk to your vendor about cmi5 support.

When the contract is nearing completion on a non-conformant system.
An ideal time to implement a cmi5 conformant LMS or system is when the existing contract is nearing the end of the contract period, it is time for a recompete, or asking for new RFPs (Request for Proposals). You can begin discussions with your LMS vendor about cmi5 support and then upgrade to that version once it becomes available.

Adopting and Implementing cmi5

The following is designed to help program managers and learning and training professionals start the process of adopting and implementing the cmi5 specification.

Beginning Vendor Conversations

Acquiring cmi5 conformant tools and systems is the first step in moving forward with implementing cmi5. To start incorporating cmi5 into your overall learning and training strategy, here are a few conversations to have with vendors.

Ask your authoring tool vendor(s) to support the cmi5 specification.
If your current authoring tools do not support cmi5, now is the time to state that cmi5 is a requirement. As more organizations start asking for support, the more likely it is that content development tools will add cmi5 support.

Ask your simulations and virtual reality vendors if they support cmi5.
If you have a cmi5 conformant LMS, ask your simulations, virtual reality, augmented reality, and other newer technologies vendors if they are able to publish learning activities as cmi5 or if cmi5 is on their roadmap. If your department doesn't currently have a cmi5 conformant LMS or system, you can still begin talks with vendors now about plans to implement cmi5 for your training program. If the vendor doesn't currently support cmi5, these talks will give them time to add support for cmi5 into their learning activities.

Ask your current LMS vendor if they support the cmi5 specification.
If you are unsure if your LMS vendor supports the specification, now is the time to start asking. They may already have support, and you can begin to utilize cmi5 learning activities. If they don't currently have support, ask if cmi5 is on their roadmap. The more organizations and companies that start asking for support, the more likely it is that vendors will adopt the specification.

Along with confirming that cmi5 support is available, you will want to validate conformance by asking for their conformance test suite results.

Adopting cmi5 Within Your Organization

To fully realize the benefits of cmi5, it is recommended that you address both the content and systems adoption in parallel.

Acquiring a cmi5 conformant LMS or system.
Acquiring a cmi5 conformant LMS with an xAPI LRS is the first step to start utilizing cmi5 and seeing the benefits of xAPI data tracking with an LMS.

Publishing new learning activities as cmi5.
If your authoring tool has the ability to publish as cmi5, start publishing courses as cmi5. Instructional designers and content authors can start publishing courses as cmi5 now, even if your LMS does not currently support cmi5. Simply publish the learning activity as cmi5 and also publish in the LMS's accepted standard. This way, you will already have courses available when you do acquire an LMS.

Deciding which existing legacy courseware you want to migrate to cmi5.
If you have a large content library, the idea of migrating that content to another standard might be daunting. It is important to note that you do not have to migrate all of your content at one time.

Here are a few ideas on which courses might be best to migrate first. When the content:

- Needs to be updated or revised.
- Benefits from advanced tracking data.
- Is accessed frequently by learners.
- Is required for a large number of learners.

Using This Guide to Start Implementing cmi5

The rest of this guide is divided into two main sections: cmi5 for Content Professionals and cmi5 for Systems Professionals. The following subsections will help you navigate to the parts that pertain most closely to your role and goals.

Chapter 5 Deep Dive into cmi5 Content

Key objectives:

- Learning more about the technical aspects of cmi5 as it relates to content
- Discovering Assignable Unit (AU) responsibilities
- Understanding how cmi5 works for content or learning activities

Who it's for:

- Program managers
- Acquisitions professionals
- Content professionals in similar roles

Chapter 6 Acquiring cmi5 Content and Content Development Tools

Key objectives:

- Understanding when to acquire cmi5 content and authoring tools
- Discovering the best practices for acquiring cmi5 content
- Identifying the key questions to ask vendors

Who it's for:

- Program managers
- Acquisitions professionals
- Content professionals in similar roles

Chapter 7 cmi5 Content Acquisition Guidance and Contracting Language Examples

Key objectives:

- Reviewing cmi5 content contracting language examples
- Avoiding contracting pitfalls
- Seeing an example Data Item Description (DID)

Who it's for:

- Program managers
- Acquisitions professionals
- Content professionals in similar roles

cmi5 for Content Professionals

Chapter 8 Building and Migrating cmi5 Content

Key objectives:

- Building cmi5 content best practices and avoiding common mistakes
- Migrating legacy SCORM content
- Using the cmi5 course templates to build or migrate content

Who it's for:

- Content developers
- Instructional designers
- Content professionals in similar roles

Chapter 9 Testing in the cmi5 Content Test Suite (CTS)

Key objectives:

- Discover the importance of validating cmi5 content in the conformance test suite
- Learning more about the CTS
- Understanding how to use the CTS

Who it's for:

- Content testers
- Content developers
- Instructional designers
- cmi5 for Systems Professionals

cmi5 for Systems Professionals

Chapter 10 Deep Dive Into cmi5 Systems Requirements

Key objectives:

- Learning more about the technical responsibilities of cmi5 systems
- Discovering the LMS launch requirements
- Understanding more about cmi5 conformant systems

Who it's for:

- Systems developers
- Systems professionals
- Program managers

Chapter 11 Acquiring cmi5 Systems

Key objectives:

- Learning when to acquire cmi5 conformant systems
- Discovering the best practices for acquiring cmi5 systems
- Understanding the key requirements and questions to ask vendors

Who it's for:

- Program managers
- Acquisitions professionals
- LMS administrators

Chapter 12 cmi5 Systems Acquisition Guidance and Contracting Language Examples

Key objectives:

- Reviewing cmi5 systems contracting language examples
- Avoiding contracting pitfalls
- Seeing an example Data Item Description (DID)

Who it's for:

- Program managers
- Acquisitions professionals
- LMS administrators

Chapter 13 Building cmi5 Systems

Key objectives:

- Understanding cmi5 systems best practices
- Using the cmi5 Player Prototype to build a system
- Testing in the cmi5 LMS Test Suite (LTS)

Who it's for:

- Systems developers
- LMS administrators
- Systems professionals in similar roles

Chapter 14 Testing in the cmi5 LMS Test Suite (LTS)

Key objectives:

- Discovering the importance of testing LMS (system) conformance
- Learning about the LTS
- Understanding how to use the LTS

Who it's for:

- LMS vendors
- Systems developers
- Systems professionals in similar roles

cmi5 for Content Professionals

In this section:

- cmi5 deep dive: Learning content
- Acquiring cmi5 content
- cmi5 content acquisition guidance and contracting language examples
- Implementing cmi5
- Building cmi5 content

Purpose of this section:

This section is for content professionals or professionals working with learning and training content, such as program managers, instructional designers, content authors, and content developers, to learn more about cmi5 content.

This section answers:

- What are the more technical aspects of cmi5?
- What are the best practices for acquiring cmi5 content?
- What contracting language should be used when acquiring content?
- How do I implement cmi5?
- What are the best practices for building cmi5 conformant content?

Chapter 5

Deep Dive Into cmi5 Content

In this chapter:

- Course structure
- Course package
- Assignable Units (AUs)
- launchParameters
- “cmi5 allowed” statements
- State API usage
- Using cmi5 with other xAPI Profiles
- cmi5 extensions

5.1 Purpose

This chapter is intended to provide a deeper dive into the content features of the cmi5 specification. The guidance in this chapter, when combined with other resources like Chapter 8 Building and Migrating cmi5 Content and the full cmi5 specification, functions as a cmi5 resource for content administrators, instructional designers, content authors, content testers, content developers, and other content professionals. The scope of this chapter is limited to the cmi5 specification only and does not address utilizing SCORM or other standards.

5.1.1 How to Use This Chapter

This chapter is designed to be used as a guide for understanding the most common content features of cmi5. For more in-depth knowledge and to review cmi5 requirements, visit the full cmi5 specification. The terms “learning activity” or “content” are used throughout the following sections to denote cmi5 content and to encompass all modalities of cmi5 content, including, but not limited to, authoring or content development tools, traditional courseware, videos, simulations, virtual reality, or augmented reality content. Assignable Unit (AU), the launchable piece of cmi5 content, is also used in accordance with the cmi5 specification. To match the terminology used in the cmi5 specification, LMS is used to describe the system that is “launching” the cmi5 content, but the system may or may not be a traditional Learning Management System. MUST and MUST NOT are capitalized when referring to a requirement of the cmi5 specification.

5.1.2 Who This Chapter Is For

This chapter is for content administrators, instructional designers, content authors, content testers, content developers, and other content professionals wanting to learn more about cmi5.

5.2 Course Package

A course package is an optionally packaged ZIP file that includes a course structure file, all Assignable Units (AUs) of the learning activity, and any additional assets or optional learning content used by AUs, such as videos or a PDF. Unlike legacy standards, a cmi5 course package does not have to contain all the resources in a single archive (usually a ZIP file). All files can be “packaged,” even by reference.

Technically, a course package is an XML file format with a course structure and may be standalone or contained in a ZIP file. When located in a ZIP file, the file MUST be named `cmi5.xml`. The course package MUST conform to https://w3id.org/xapi/profiles/cmi5/v1/CourseStructure.xsd and be available locally in v1/CourseStructure.xsd.[1]

How cmi5 course packages differ from SCORM:

- **Supports 1,000+ AUs**. A cmi5 conformant LMS must be able to import and support a course with at least 1,000 AUs. (See section 4.4 Assignable Units)
- **Remote content.** The course package can contain media or other content that is stored remotely. In this type of learning activity, the course structure is still imported into the LMS, but URLs are used to access remote content or resources that reside in other domains.
- **Clearly defined satisfaction criteria.** Satisfaction criteria, or moveOn criteria, is more clearly defined and could be: “Completed”, “Passed”, “CompletedAndPassed”, or “CompletedOrPassed”. The LMS may override the moveOn criteria in cmi5.[2]
- **Control over the learner presentation.** The course structure can indicate that it must be launched in an independent browser context (popup, redirect to content directly, etc.) from the LMS.

5.3 Course Structure

The course structure is a list of AUs and their launch parameters with an implied sequence that represents a course or learning activity. Like with the legacy SCORM standard, learning activities are imported into an LMS. cmi5 differs from SCORM in that the actual AU files might not be imported into the LMS, only the course “structure” is imported. AU(s) can reside anywhere, including mobile applications, content distribution networks, etc.

In a course package saved as a single archive in a ZIP file, the course structure is saved as an XML file called `cmi5.xml`, which wraps the pieces of grouped content (blocks) or launchable AUs.

5.4 Assignable Units (AUs)

Assignable Units (AUs) are separately launchable pieces of content that include the concepts of completion, pass/fail, score, and duration, and every learning activity is required to have at least one AU. The AU is the unit of management and tracking that collects learner data and sends the data to the LMS.

The AU's responsibilities are as follows:

- Parse the parameters from the launching environment to determine where the LMS location is and initiate communication with the LMS.
- Act as a "client," sending and receiving messages using the defined transport mechanism(s) and associated commands as prescribed in the cmi5 specification.
- Format all data according to the defined data types and vocabularies that are defined in the cmi5 specification.
- Send a "Terminate" statement prior to terminating the AU's execution.[3]

Further, AUs are required to do the following per the cmi5 specification:

- Implement runtime communication as defined in the xAPI Specification.
- Implement Statement API requirements as defined in the AU Statement API requirements listed in Section 7.1.
- Implement the runtime launch interface as defined in the content launch mechanisms in Section 8.
- Implement State API requirements in the cmi5 specification as defined in the xAPI state data model in Section 10.
- Implement Profile API requirements in the cmi5 specification as defined in the xAPI agent profile data model in Section 11.

5.4.1 Blocks

Blocks consist of nested groups of AUs that can be nested within another block of AUs. The cmi5 specification does not limit the number of blocks within a course package, and cmi5 conformant LMSs must be able to support learning activities with thousands of AUs.

Block metadata contains:

- **ID.** A globally unique IRI to identify the block in xAPI requests made by the LMS. This ID MUST be unique within the course structure.
- **Title.** A descriptive title for the block of AUs.
- **Description.** A detailed written description of what the block contains.
- **Objectives.** A listing of objectives referenced by this block. Objectives are not a requirement of the cmi5 specification.[4]

5.4.2 Verbs and Ordering Rules

The cmi5 specification defines nine xAPI Verbs and ordering rules for the Verbs. Following the specification, all statements in this subsection refer to "cmi5 defined" statements unless otherwise noted. "cmi5 allowed" statements are NOT subject to the usage rules listed below (see Section 5.6 "cmi5 allowed" statements).[5]

Figure 5-1: About the Nine "cmi5 defined" Verbs

Launched	
Description	Indicates that the AU was launched by the LMS.
AU Obligations	No AU obligations.
LMS Obligations	The LMS MUST use this verb in a statement recorded in the LRS before launching an AU.
Initialized	
Description	Indicates that the learner's cmi5 session has been fully initialized. The "Initialized" statement MUST follow, within a reasonable period of time, the "Launched" statement created by the LMS.
AU Obligations	The AU MUST use "Initialized" in the first statement (of any kind) in the AU session.
LMS Obligations	No LMS obligations.
Completed	
Description	Indicates the learner viewed or did all of the relevant activities in an AU presentation. The use of the "Completed" verb indicates progress of 100%.
AU Obligations	The AU MUST send a statement containing the "Completed" verb when the learner has experienced all relevant material in an AU.
LMS Obligations	The LMS MUST use "Completed" statements based on the moveOn criteria for the AU as provided in the LMS Launch Data.
Passed	
Description	Indicates the learner attempted and succeeded in a judged activity in the AU.
AU Obligations	The AU MUST send a statement containing the "Passed" verb when the learner has attempted and passed the AU. If the "Passed" statement contains a (scaled) score, the (scaled) score MUST be equal to or greater than the "masteryScore" indicated in the LMS Launch Data.
LMS Obligations	The LMS MUST use "Passed" statements based on the moveOn criteria for the AU as provided in the LMS Launch Data.

Failed	
Description	Indicates the learner attempted and failed in a judged activity in the AU.
AU Obligations	The AU MUST send a statement containing the “Failed” verb when the learner has attempted and failed the AU. If the “Failed” statement contains a (scaled) score, the (scaled) score MUST be less than the “masteryScore” indicated in the LMS Launch Data.
LMS Obligations	No LMS obligations.
Abandoned	
Description	Indicates that the AU session was abnormally terminated by a learner's action (or due to a system failure).
AU Obligations	No AU obligations.
LMS Obligations	In the absence of a “Terminated” statement, the LMS MUST make the determination if an AU abnormally terminated a session by monitoring new statement or State API requests made for the same learner/course registration for a different AU. The LMS MUST record an “Abandoned” statement on behalf of the AU indicating an abnormal session termination.
Waived	
Description	Indicates that the LMS has determined that the AU requirements were met by means other than satisfying the AU.
AU Obligations	No AU obligations.
LMS Obligations	The LMS MUST use this verb in a statement recorded in the LRS when it determines that the AU may be waived. A statement containing a “Waived” verb MUST include a “reason” in the extension property of the Statement Result. The LMS MUST generate a unique session ID for the statement containing a “Waived” verb and MUST NOT issue any other statements (except for statements with the “Satisfied” verb) using that session ID.
Terminated	
Description	Indicates that the AU was terminated by the Learner and that the AU will not be sending any more statements for the launch session.
AU Obligations	The AU MUST send a statement containing the “Terminated” verb. This statement MUST be the last statement (of any kind) sent by the AU in a session.
LMS Obligations	The LMS MUST use the “Terminated” statement to determine that the AU session has ended. Upon receiving a “Terminated” statement, the LMS MUST wait a specified period of time (defined by the LMS implementation), after which it MUST reject statements (of any kind) for the AU session.

Satisfied	
Description	Indicates that the LMS has determined that the learner has met the moveOn criteria of all AUs in a block or has met the moveOn criteria for all AUs in the course.
AU Obligations	No AU obligations.
LMS Obligations	The LMS MUST use the “Satisfied” statement when the learner has met the moveOn criteria of all AUs in a block. In this statement, the LMS MUST use the block object per Section 9.4. The LMS MUST use the same block object for all “Satisfied” statements that refer to that block. The LMS MUST also use the “Satisfied” statement when the learner has met the moveOn criteria for all AUs in a course. In this statement, the LMS MUST use the course object per Section 9.4. The LMS MUST use the same course object for all “Satisfied” statements that refer to that course. The LMS MUST use the session ID from the AU launch for all “Satisfied” statements triggered as a result of an AU launch session. The LMS MUST generate a unique session ID for all “Satisfied” statements triggered outside of an AU launch session.

AU Verb Ordering Rules

The cmi5 specification states that AUs MUST use the above cmi5 Verbs that are indicated as mandatory in other sections of the cmi5 specification. For all requirements listed, including language involving order, the order is determined by the value of the timestamp property.

AU Verb ordering rules within an AU session are:

- Verbs MUST NOT be duplicated (in “cmi5 defined” statements).
- More than one of the sets of {“Passed”,”Failed”} verbs MUST NOT be used (in “cmi5 defined” statements).
- The “Initialized” verb MUST be the first statement (cmi5 allowed or defined).
- The “Terminated” verb MUST be the last statement (cmi5 allowed or defined).

AU Verb ordering rules within a registration (per AU) are as follows:

- Exactly zero or one “Completed” “cmi5 defined” statement MUST be used per registration.
- Exactly zero or one “Passed” “cmi5 defined” statement MUST be used per registration.
- A “Failed” statement MUST NOT follow a “Passed” statement (in “cmi5 defined” statements) per registration.

To realize the full potential of xAPI tracking data, AUs may use additional verbs not listed in the specification.

LMS Verb Ordering Rules

The LMS MUST record and provide reporting for all “cmi5 defined” and “cmi5 allowed” statements that are not being rejected regardless of the Verbs used in statements sent by AUs.

LMS Verb ordering rules are as follows:

- LMS may issue multiple “Satisfied” statements in a session.
- The LMS SHOULD NOT issue multiple statements with “Satisfied” for the same block or course within a course registration for a given learner.
- LMS MUST NOT issue more than one “Abandoned” statement for a session.
- LMS MUST NOT issue more than one “Waived” statement per session and MUST not issue more than one “Waived” statement per registration per AU.

5.4.3 Other AU Features

“masteryScore”

The “masteryScore” is an optional mechanism to control the course’s behavior relative to determining passed or failed status. If the AU has scoring and the course structure has a “masteryScore”, this value (see Section 10.2.4 of the cmi5 specification) MUST be used by the AU as an override behavior to determine the passed or failed status of the registration for that AU. This value is included in the LMS Launch Data activity state, which the AU must *fetch*. The “masteryScore” must be between 0.0 and 1.0. The AU is responsible for sending a “Passed” statement if the scaled score is greater than or equal to the “masteryScore” or a “Failed” statement if the scaled score is less than the “masteryScore”. The AU MUST also include the result’s score block in either statement. The AU MUST also include the “masteryScore” value provided by the LMS in the context of an extension for the “Passed”/”Failed” statements it makes based on the “masteryScore”.

moveOn Values

moveOn Values are used by the LMS to determine if an AU has been sufficiently satisfied for the purposes of determining overall course satisfaction or determining if prerequisites were met for other activities.

moveOn Values are:

- **“Passed”:** The LMS considers the AU satisfied when it receives a statement with the verb “Passed”.
- **“Completed”:** The LMS considers the AU satisfied when it receives a statement with the verb “Completed”.
- **“CompletedAndPassed”:** The LMS considers the AU satisfied when it receives statements with the verbs “Completed” and “Passed”.
- **“CompletedOrPassed”:** The LMS considers the AU satisfied when it receives a statement with either of the verbs “Completed” or “Passed”.
- **“NotApplicable”:** The LMS considers the AU satisfied.

A block is considered satisfied for prerequisites and sequencing when all member AUs in a block are satisfied. The course is considered satisfied for prerequisites or credit in relation to other courses or curricula when all member AUs and blocks are satisfied.

Objects

An Object MUST be present, as specified in this subsection, in all “cmi5 defined” statements, and an Object has additional requirements in the cmi5 specification. Except for statements with the “Satisfied” verb, the Object in a “cmi5 defined” statement represents the AU. When the Object is the AU, the value of the Object’s “id” property for a given AU MUST match the activityId defined in the launch URL. In “Satisfied” statements, the Object represents a block or learning activity.

5.5 launchParameters

cmi5 is an xAPI Profile that defines “launching” with an LMS. The cmi5 specification defines launchParameters as follows.

The launchParameters are defined in the cmi5 course structure. The LMS MUST include the launchParameters in the State API Document if the launchParameters were defined by the course designer in the course structure.

The launchParameters value written in the State API Document MAY be different than the one in the course structure (e.g. based on content vendor options that may be used by LMS administrative users). The AU SHOULD get the launchParameters value from the State API Document if the launchParameters were defined in the course structure.[6]

5.6 “cmi5 Allowed” Statements

“cmi5 allowed” statements are xAPI Statements that include the session ID and potentially additional information included in the launch data but are otherwise custom to an AU. “cmi5 allowed” statements may use any verb and cmi5 context template (but NOT including the cmi5 category ID).

“cmi5 allowed” statements posted by the AU MUST occur between cmi5 statements using the “Initialized” verb and the “Terminated” verb. “cmi5 allowed” statements are not considered in “cmi5 defined” session management and satisfaction rules.[7]

5.7 State API Usage

In some instances, the AU may need to have the ability to store data across sessions (or other occurrences). It is recommended that this be done via the State API. For example, the SCORM concepts of bookmarking/location and suspend data would likely be implemented via the State API when adhering to the cmi5 specification.

5.8 Using cmi5 With Other xAPI Profiles

cmi5 was designed to be able to work in conjunction with other xAPI Profiles, including the Video, Audio, and Serious Games Profiles. Not only was cmi5 designed specifically with this purpose, it is recommended that content creators and developers use common xAPI Profiles as needed to ensure that learning activities maintain plug and play interoperability and data portability. The DoDI 1322.26 fungible assets specifically states that, where applicable, the use of multiple xAPI Profiles is encouraged.[8] A complete list of known xAPI Profiles can be accessed from the ADL Vocabulary Server.

5.9 cmi5 Extensions

The cmi5 specification does not address Sequencing in the same way that SCORM did. To address this, the cmi5 Working Group developed two methods for extending features, which are not part of the specification but are compatible. If the following features apply to your use case, it is recommended that they be used to ensure interoperability:

- requires – Defines a specified list of AUs that must have met their moveOn criteria before a given AU is accessible by the learner. More about the requires extension: https://aicc.github.io/CMI-5_Spec_Current/extensions/requires.html
- collateralCredit – Defines a list of blocks and AUs that the LMS MUST issue “Waived” statements when a given AU is satisfied. More about the requires extension: https://aicc.github.io/CMI-5_Spec_Current/extensions/collateralcredit.html

Chapter 6

Acquiring cmi5 Content and Content Development Tools

In this chapter:

- When to acquire cmi5 content
- Best practices for acquiring cmi5 content
- Acquiring a content development (authoring) tool
- What to ask vendors

6.1 Purpose

This chapter is intended to provide acquisition guidance and best practices for acquiring cmi5 content and content development (authoring) tools. This chapter, in conjunction with the following chapter providing example contracting language, functions as a resource for new content tool acquisitions that require cmi5. The guidance in this chapter, when combined with other sources, such as the following chapter, should provide sufficient cmi5 recommendations for acquisition professionals, program managers, and similar roles to include in new cmi5 content procurement contracts. The scope of this chapter is limited to the cmi5 specification only and does not address requirements for content utilizing SCORM or other standards. This chapter presents a range of possible considerations for acquiring content and tools, and as the standard evolves and progresses, the considerations may change over time.

6.1.1 How to Use This Chapter

This section should be interpreted as the authoritative guide for acquiring cmi5 conformant content. This chapter covers the cmi5 content and authoring tools acquisition requirements and includes information on when to acquire cmi5 content, best practices for content acquisitions, best practices for authoring tool acquisitions, and what to ask vendors.

The terms “learning activity” or “content” are used throughout the following sections to denote cmi5 content and to encompass all modalities of cmi5 content, including, but not limited to, authoring or content development tools, traditional courseware, videos, simulations, virtual reality, or augmented reality content. Assignable Unit (AU), the launchable piece of cmi5 content, is also used in accordance with the cmi5 specification. To match the terminology used in the cmi5 specification, LMS is used to describe the system that is “launching” the cmi5 content, but the system may or may not be a traditional Learning Management System.

6.2 When to Acquire cmi5 Content

This chapter specifies only cmi5 contracting requirements, but it's important for acquisition professionals to determine when cmi5 is a contract requirement or when cmi5 would be the e-learning standard.

Acquire cmi5 content if learning activities will be launched from an LMS and one or more of the following are true:

- **Following Option 2 in the DoDI 1322.26 fungible references.**[1] DoDI 1322.26 states that DoD Components shall begin migrating toward the use of xAPI standards with acquiring and maintaining a cmi5 conformant LMS and an xAPI LRS as one of the xAPI implementation paths.
- **Capturing more detailed learning data.** cmi5 is the preferred standard when program managers, content authors, or instructional designers desire to see more detailed data about what happens inside the course or learning activities beyond what the SCORM standards support.
- **Incorporating xAPI enabled activities.** cmi5 is the preferred xAPI Profile when incorporating xAPI enabled activities that will be launched from an LMS. Using an xAPI Profile other than cmi5, such as the Launch Profile or SCORM Profile, could cause problems with data portability and semantic interoperability if data are shared across federated platforms.
- **Implementing new training modalities.** Legacy SCORM standards are not extensible enough to track newer training modalities, such as simulations, virtual reality, augmented reality, and non-browser based applications. cmi5 is the preferred specification to use when incorporating these newer modalities into training programs and launching from an LMS.
- **Acquiring learning activities that are hosted outside of an LMS.** Unlike SCORM, cmi5 supports Content as a Service (CaaS) models of delivery so that content can be stored outside of an LMS.

6.3 Best Practices for Acquiring cmi5 Content

The following content best practices should be considered when acquiring cmi5 learning activities. Example contracting language for acquiring cmi5 content is provided in the next chapter of this best practices guide.

6.3.1 Passed the cmi5 Content Test Suite (CTS)

All off-the-shelf or acquired content should be thoroughly tested in the cmi5 Content Test Suite (CTS). The full course should be tested to ensure a passing result from the CTS. Once tested, the vendor will need to provide the test reports for all expected use case scenarios, such as learner experiences sufficient content to mark completion or learner attempts a scored activity, to prove that the course is conformant to the cmi5 specification.

6.3.2 Support the SHOULD and SHOULD NOT Aspects of the cmi5 Specification

The following SHOULD and SHOULD NOT aspects need to be supported in all learning activities. Failing to incorporate one of the following SHOULD or SHOULD NOT aspects may cause interoperability concerns. Links to the corresponding part of the cmi5 specification[2] are included for reference.

7.1.2 Last Statement Call

Once the AU has determined that the session will end (e.g., by user action, timeout, or some other means) the AU SHOULD issue a “Terminated” statement.

8.1.2 Content Launch Mechanisms - *fetch*

The AU MUST get the *fetch* value from the query string. The AU MUST make an HTTP POST to the *fetch* URL to retrieve the authorization token as defined in Section 8.2. The AU MUST place the authorization token in the Authorization headers of all HTTP requests made to the endpoint using the xAPI. The AU SHOULD NOT make more than one post to the *fetch* URL.

8.2.1 Authorization Token *Fetch* URL

The AU SHOULD NOT attempt to retrieve the authorization token more than once.

In addition, the learning activity will also need to support any SHOULD and SHOULD NOT aspects that apply to that specific learning activity type.

6.3.3 General cmi5 Content Acquisition Best Practices

Includes relevant IRI title and descriptions

The learning activity follows the best practice of using unique titles and descriptions that are relevant to, and descriptive of, the content.

Provides registration ID in all statements

The learning activity includes the registration ID in all statements.

Documents Activity ID inventory list

The acquisition documentation includes an inventory of Activity information in order to avoid ID collisions and multiple IDs for the same activity. The inventory list document should include (at a minimum):

- Activity IDs (IRI)
- Activity names
- Activity descriptions
- Activities that are the Object of an xAPI Statement or contextActivities in an xAPI statement

Uses "Progressed" verb for indicating progress during a session

For recording progress during a session, the learning activity uses a "cmi5 allowed" statement with the "Progressed" verb and a progress extension in the result (see Section 9.5.5.1). Once the learner reaches 100%, it is recommended that a "cmi5 defined" "Completed" statement be issued instead.

Uses objectives

Objectives are defined for the course structure, but there is no language in the specification concerning their usage in statements. If an AU is using objectives in statements, then the best practice is to add the objective (with the same objective ID provided in the course structure) to the context activities "grouping" property as an activity type of https://adlnet.gov/expapi/activities/objective from the ADL vocabulary.

Includes AU “masteryScore” (if applicable to the learning activity)

If the LMS issues a “masteryScore”, then the AU will respond in the following ways:

- If the AU has no notion of scoring, it will not issue “Passed” or “Failed” statements.
- If the AU does not have scoring or the learner does not meet the “masteryScore”, then the AU MUST NOT issue a “Passed” statement.
- The AU can also refuse to execute because the “masteryScore” issued is inconsistent with the learning design of the AU. In this situation, the AU will inform the learner why it cannot execute.

Specifies a moveOn criterion in the course structure for each AU

The learning activity always specifies a moveOn criteria for each AU.

Closes the browser window to exit in the absence of returnURL

The AU closes the Window or directs the learner to manually close the browser window if the LMS does not provide a “returnURL”.

Uses a derived activity ID for “cmi.interaction” statements

The AU uses an Activity ID derived from the AU’s Activity ID in accordance with the cmi5 content best practices.

Addresses persistent AU Session State

The AU uses non-volatile storage (local to the AU) to preserve the *fetch* URL token retrieval, which can only be called once in session, and all “cmi5 defined” statements that can only be sent once in a session.[3]

6.4 Acquiring a Content Development (Authoring) Tool

The following best practices should be considered when acquiring content development (authoring) tools that support the cmi5 specification. If an authoring tool does not follow these best practices, learning activities created using the tool may experience conformance and interoperability issues. Example contracting language for acquiring an authoring tool is provided in the next chapter of this best practices guide.

Supports ALL of the MUST and MUST NOT aspects of the cmi5 specification

It is required that the tool supports all of the MUST and MUST NOT requirements of the cmi5 specification. If the tool does not support a MUST or MUST NOT requirement, then the tool will be considered non-conformant to the specification and interoperability issues may arise.

Supports the SHOULD and SHOULD NOT aspects of the cmi5 specification

It is a best practice that the content development tool will support the use of ALL of the SHOULD or SHOULD NOT aspects of the cmi5 specification as described in the specification version Quartz (or latest version).

Courses created with the tool have passed the cmi5 Content Test Suite (CTS)

A best practice for the authoring tool is to show that a variety of learning activities with different course structures were created using the tool, and the learning activities have been successfully tested and passed the CTS.

Supports complex course structures

The authoring tool has the ability to support complex course structures, including multiple objectives, multiple blocks with nesting, multiple AUs in a block, learning activities with a “masteryScore”, etc.

Supports persistent AU session state

There are certain situations, such as browser window refresh, that may inadvertently cause cmi5 errors related to operations that must occur only once in a session. To prevent such issues, AUs should be designed to use non-volatile storage (local to the AU) to preserve the state of the following operations that have been performed during the session:

- *Fetch* URL token retrieval (which can only be done once per session).
- All “cmi5 defined” statements that can only be sent once in a session.

6.5 What to Ask Vendors

The following section is designed to help acquire cmi5 conformant learning activities and acquire content development (authoring) tools by providing questions for the vendor to answer in a Request for Proposal (RFP), Data Item Description (DID) deliverable, Contract Data Requirements List (CDRL), or other procurement contract.

6.5.1 Content Providers

Conformance to the cmi5 specification.

- Does the learning activity conform to the cmi5 specification version Quartz or later?
- Does the learning activity follow all of the MUST and MUST NOT aspects of the cmi5 specification version Quartz or later?
- Does the learning activity follow the applicable SHOULD and SHOULD NOT aspects of the cmi5 specification version Quartz or later?
- How were the IRIs produced? Are the IRIs consistent?

Testing in the cmi5 Content Test Suite.

- Has the learning activity (course) passed the cmi5 Content Test Suite?
- Was the learning activity, in its entirety, tested?
- Will you provide all applicable test reports?

Specific learning activities.

- **AU "masteryScore".** If the LMS issues a "masteryScore", does the AU respond in the appropriate way as defined in the cmi5 specification?
  - If the AU has no notion of scoring, it will not issue "Passed" or "Failed" statements.
  - If the AU does not have scoring or the learner does not meet the "masteryScore", then the AU MUST NOT issue a "Passed" statement.
  - The AU can also refuse to execute because the "masteryScore" issued is inconsistent with the learning design of the AU. In this situation, the AU should inform the learner why it cannot execute.
- **Supports multiple languages.** If the AU supports multiple languages, does the AU display in the preferred language preference order of the user?

- **Browse launchMode.** If the AU uses the Browse launchMode, does the AU allow the learner to 'look around' without judgment?
- **Review launchMode.** If the AU uses the Review launchMode, does the AU allow the learner to "revisit/review" already completed material?
- **Statements that include duration.** If the AU statement includes duration, does the "Terminated," "Completed," "Passed," or "Failed" statement follow the SHOULD and SHOULD NOT aspects as defined in the cmi5 specification?

Following the cmi5 best practices.

- Do IRIs, courses, blocks objectives, and AUs each contain unique titles and descriptions that are relevant and descriptive of the learning activity?
- Is the registration ID included in all statements?

6.5.2 Content Development (Authoring) Tool Vendors

Conformance to the cmi5 specification.

- Does the tool support the entirety of the cmi5 specification version Quartz or later?
- Does the tool support all of the MUST and MUST NOT aspects of the cmi5 specification version Quartz or later?
- Does the tool support all of the SHOULD and SHOULD NOT aspects of the cmi5 specification version Quartz or later?
- Does the tool have the capability to support moveOn criteria, including "NotApplicable," "Passed," "Completed," "CompletedOrPassed," and "CompletedAndPassed"?
- Does the tool allow for entering and adjusting IRIs for various activities and AU sub-objects?

Testing in the cmi5 Content Test Suite.

- Has a learning activity (content) been created with the tool and then tested in the cmi5 Content Test Suite? Did the learning activity pass cmi5 Content Test Suite?
- Following the cmi5 best practices.
- Does the tool support scaled and raw scores? For example, ranges that aren't strictly a minimum of 0 and a maximum of 100; a learner could score a maximum over 100 and receive a correct scaled score.
- How does the tool handle error conditions as well as network connection disruptions, re-launches, and/or re-entrant sessions?

Chapter 7

cmi5 Content Acquisition Guidance and Contracting Language Examples

In this chapter:

- cmi5 content contracting language examples
- Contracting pitfalls to avoid
- Example Data Item Description (DID) template for cmi5 content

7.1 Overview

This section provides cmi5 contracting requirements and contracting language examples for acquiring cmi5 enabled authoring tools or cmi5 conformant learning activities. This section is intended to provide contracting guidelines for acquisition professionals to use as a resource for procuring content, content development tools, such as authoring tools, or any content development contracts that require cmi5. The example language in this document can be reused and augmented with additional contractual requirements as necessary.

In addition, an example Data Item Description (DID) template for cmi5 data is proposed in Section 7.4. Since there are currently no official DIDs for cmi5 data, this document serves as a first draft that could be submitted for procuring cmi5 content.

Department of Defense Instruction (DoDI) 1322.26 fungible references lists cmi5 (Option 2) as one of the standards and specifications that DoD organizations should use when acquiring distributed learning solutions. The requirements in this section should provide a sufficient foundation for cmi5 contracting requirements for acquisition professionals to include in new cmi5 content development contracts. The scope of this chapter is limited to only cmi5 and does not address contracting requirements for content utilizing SCORM or other standards.

7.1.1 How to Read This Section

This section should be interpreted as the authoritative contracting requirements for acquiring cmi5 conformant content.

There are two levels of obligation with regards to cmi5 conformance requirements identified by the terms herein. Since these cmi5 conformance requirements are intended to be reused as part of the contract requirements in a Request for Proposal (RFP), Data Item Description (DID) deliverable, Contract Data Requirements List (CDRL), or other procurement contract, the use of “MUST” or “MUST NOT” and “SHOULD” or “SHOULD NOT” are only used when referring to the cmi5 specification and “SHALL” or “SHALL NOT” are used as part of the contract language. This is done intentionally in order to avoid confusion between what is required as part of the cmi5 specification and what is required of the contractor. Not following these recommendations could risk interoperability and/or lead to cmi5 deliverables not being accepted.

1. **SHALL OR SHALL NOT.** If a product fails to implement a SHALL or SHALL NOT requirement of the contract and/or fails to implement a MUST or MUST NOT requirement of the cmi5 specification, the product is considered non-conformant to the requirements.
2. **WILL.** Used to indicate the Government will complete a task.

The terms “learning activity” or “content” are used throughout the contracting language to denote cmi5 content and to encompass all modalities of cmi5 content. Modalities include, but are not limited to, authoring or content development tools, traditional courseware, videos, simulations, virtual reality, or augmented reality content. Assignable Unit (AU), the launchable piece of cmi5 content, is also used in the contracting in accordance with the cmi5 specification. To match the terminology used in the cmi5 specification, LMS is used to describe the system that is “launching” the cmi5 content, but the system may or may not be a traditional Learning Management System.

7.2 cmi5 Content Contracting Language Examples

7.2.1. General cmi5 Requirements and Example cmi5 Contracting Language

This chapter specifies only cmi5 contracting requirements, but it’s important for acquisition professionals to determine when cmi5 is a contract requirement. The distinguishing factors for when cmi5 should be contracted are as follows:

Acquire cmi5 content if content is launched with a Learning Management System (LMS) and ANY of the following contracting requirements for e-learning content are true:

- Content utilizes xAPI.
- Must have better resolution into the data and associated analytics.
- Desired data collection surpasses the data collected as defined by the SCORM standards.
- Content is delivered on any device.
- Content resides outside of the LMS server domain.

The following general requirements and example contracting language should be included in all cmi5 content acquisitions. This language provides a foundation for more detailed requirements provided by specific content types. The examples below can be used without modification, but it is expected that updates may be required based on specific requirements of the cmi5 content. Additionally, adherence to the cmi5 specification should be referenced throughout the acquisition process for any type of training and education system. These include Requests for Information, Sources Sought Notifications, Statements of Work, and other types of requests for solutions.

Requirement	Contracting Language Example
cmi5 Specification Conformance	The Contractor SHALL develop the cmi5 content such that it is conformant to the cmi5 specification version Quartz (or latest version) by adhering to the cmi5 specification and being validated in the cmi5 Content Test Suite.
Record of cmi5 Specification Conformance	The Contractor SHALL provide verification that content has been tested in the cmi5 Content Test Suite by providing the test reports for all expected use case scenarios, such as learner experiences sufficient content to mark completion, learner attempts a scored activity, etc.
Activity ID Inventory List	The Contractor SHALL deliver an inventory of Activity information to avoid ID collisions and multiple IDs for the same activity. The inventory list document SHALL contain, at a minimum, the activity IDs (IRI), activity names, and activity descriptions. The inventory list document SHALL contain all activities that are the Object of an xAPI Statement or contextActivities in an xAPI Statement.
Include Reporting Characteristics for the Learning Activity	The Contractor SHALL deliver the reporting characteristics for the learning activity. The Contractor SHALL provide all reporting data characteristics for the learning activity, such as time, completion, pass/fail, score, and individual question responses.
Content Delivery	The Contractor SHALL deliver a Content Structure.
Last Statement Call	The Contractor SHALL develop content so that once the AU has determined that the session will end (e.g., by user action, timeout, or some other means) the AU SHOULD issue a “Terminated” statement.
Content Launch Mechanisms - *fetch*	The Contractor SHALL develop the content in such a way that the AU SHOULD NOT make more than one post to the *fetch* URL.

Requirement	Contracting Language Example
Authorization Token *fetch* URL	The Contractor SHALL develop the content in such a way that the AU SHOULD NOT attempt to retrieve the authorization token more than once.
Conformance to the SHOULD and SHOULD NOT Aspects of the cmi5 Specification	The Contractor SHALL develop the content in such a way that if the learning activity includes any of the SHOULD or SHOULD NOT aspects of the cmi5 specification, including the ones listed herein and the ones that are not listed in the contracting requirements, then the Contractor SHALL implement the specified AU SHOULD and/or SHOULD NOT as described in the specification.
Specifies a moveOn Criterion in the Course Structure for Each AU	The Contractor SHALL develop the content in such a way that when creating a course structure, a moveOn criteria is always specified for each AU.

7.2.2 cmi5 Requirements and Example cmi5 Contracting Language for Acquiring Content Development Tools

The following requirements and example contracting language should be included in all cmi5 content development tool acquisitions, such as authoring tools. The examples below can be used without modification, but it is expected that updates and modifications may be required.

Requirement	Contracting Language Example
Content Development Tool is cmi5 Conformant	The Contractor SHALL develop the content development tool to be in accordance with the cmi5 specification, including, but not limited to, adhering to all of the MUST and MUST NOT aspects of the cmi5 specification version Quartz (or latest version).
Content Development Tool Allows for SHOULD and SHOULD NOT Aspects of cmi5 Specification	The Contractor SHALL develop the content development tool in such a way that it conforms to and supports the use of ALL of the SHOULD or SHOULD NOT aspects of the cmi5 specification as described in the specification version Quartz (or latest version).

7.2.3 cmi5 Requirements and Example cmi5 Contracting Language Necessary for Specific Content Types

The following requirements and example contracting language may be included in contracts with specific use cases. The examples below can be used without modification, but it is expected that updates and modifications may be required based on specific requirements of the cmi5 content.

Requirement	Contracting Language Example
Document Properties - launchParameters	The Contractor SHALL develop the content in such a way that the AU SHOULD get the launchParameters value from the State API Document if the launch parameters were defined in the course structure.
xAPI Agent Profile Data Model - languagePreference	The Contractor SHALL develop the content in such a way that if the AU supports multiple languages and the supports the user's preferred language, then the AU SHOULD display in the language preference order of the user. If the AU does not support multiple languages, or if no languagePreference is specified in the Agent Profile, it may display in its default language.
AU Statements That Include Duration	When an AU statement includes duration, the contractor SHALL develop the content in such a way that it conforms to the following: **"Terminated" Statement** The AU MUST include the "duration" property in "Terminated" statements. The AU SHOULD calculate duration for "Terminated" statements as the time difference between the "Initialized" statement and the "Terminated" statement. The AU may use other methods to calculate the duration based on criteria determined by the AU. **"Completed" Statement** The AU MUST include the "duration" property in "Completed" statements. The AU SHOULD calculate duration as the time spent by the learner to achieve completion status. **"Passed" Statement** The AU MUST include the "duration" property in "Passed" statements. The AU SHOULD calculate duration as the time spent by the learner to attempt and succeed in a judged activity of the AU. **"Failed" Statement** The AU MUST include the "duration" property in "Failed" statements. The AU SHOULD calculate duration as the time spent by the learner to attempt and fail in a judged activity of the AU.

Requirement	Contracting Language Example
Extensions - Progress	The Contractor SHALL develop the content in such a way that the AU SHOULD NOT set a progress value in a “Completed” statement or if it has previously issued a “Completed” statement for the AU in the current registration.
Document Properties - *launchMode*	The Contractor SHALL develop the content in such a way that the launch mode conforms to the following and is determined by the LMS. There are three possible values: • **Normal** - Indicates to the AU that satisfaction-related data MUST be recorded in the LMS using xAPI Statements. • **Browse** - Indicates to the AU that satisfaction-related data MUST NOT be recorded in the LMS using xAPI Statements. When Browse mode is used, the AU SHOULD provide a user experience that allows the user to “look around” without judgment. • **Review** - Indicates to the AU that satisfaction-related data MUST NOT be recorded in the LMS using xAPI Statements. When Review mode is used, the AU SHOULD provide a user experience that allows the user to “revisit/review” already completed material.
Document Properties - entitlementKey	The Contractor SHALL develop the content in such a way that the AU SHOULD use this data in combination with other data provided from the LMS to determine entitlement.
Document Properties - *alternate*	The Contractor SHALL develop the content in such a way that the AU SHOULD use this data in combination with other data provided from the LMS to determine entitlement.
AU “masteryScore”	The Contractor SHALL develop the content in such a way that if the LMS issues a “masteryScore”, the AU responds in the following ways: • If the AU has no notion of scoring, it will not issue “Passed” or “Failed” statements. • If the AU does not have scoring or the learner does not meet the “masteryScore” then the AU MUST NOT issue a “Passed” statement. • The AU can also refuse to execute because the “masteryScore” issued is inconsistent with the learning design of the AU. In this situation, the AU should inform the learner why it cannot execute.

7.3 Contracting Pitfalls to Avoid

The following general contracting pitfalls should be considered in all cmi5 content acquisitions. This language provides a foundation for areas to be aware of when acquiring content, but contracting pitfalls are not limited to only these areas. It is expected that updates may be required to the below points based on specific requirements of the cmi5 content and as cmi5 content evolves.

Not clearly setting vendor expectations.

Professionals looking to acquire cmi5 conformant content will need to set clear expectations for the vendor in terms of the content being cmi5 conformant (tested in the cmi5 Content Test Suite), that all of the expected reporting characteristics for the learning activity are included, and clearly defining the expectations of the content, for example. Acquisitions professionals will need to be specific in the contracting language as being conformant to the cmi5 specification may not be sufficient for the desired use case.

Failing to have the Contractor define other xAPI Profiles used in the learning activity.

If the learning activity utilizes another xAPI Profile, such as the Video, Audio, or Serious Games Profiles, the Contractor will need to specify which xAPI Profile(s), and the Contractor will need to adhere to the guidance specified for that Profile.

Course tests do not demonstrate that the course has been exercised adequately.

If a course test completes by navigating through one or two slides, the tester should still demonstrate that the full use of the course does not cause conformance to fail. Each course acquired should be thoroughly tested in the cmi5 Content Test Suite.

Failing to implement the SHOULD and SHOULD NOT aspects of the specification correctly.

If the learning activity includes any of the optional SHOULD or SHOULD NOT aspects of the cmi5 specification, the Contractor will need to implement the aspects in accordance with the specification in order to maintain conformance. The cmi5 Content Test Suite does not currently test for SHOULD and SHOULD NOT aspects of the specification.

7.4 Example Data Item Description (DID) Template for cmi5 Content

DATA ITEM DESCRIPTION

Title: cmi5 Data

Number: TBD	**Approval Date:** TBD
AMSC Number: TBD	**Limitation:** TBD
DTIC Applicable: TBD	**GIDEP Applicable:** TBD
Preparing Activity: TBD	**Project Number:** TBD
Applicable Forms: TBD	

Use/relationship: cmi5 Data will be used for describing, reporting, and visualizing one's experience or performance within a formal or informal learning activity. Additional information is available at: https://adlnet.gov/projects/cmi5-specification/.

a. Information to be recorded through cmi5 data will include interactions and usage with e-learning content, performance support applications, and other related types of training content.

b. This DID contains the format, content, and intended use information for the data product resulting from DoD Instruction 1322.26 (Distributed Learning) and the work task described by the contract, and is applicable to the acquisition of any military learning and training content that requires cmi5.

Requirements:

1. Reference documents. The applicable issue of the documents cited herein, including their approval dates and dates of any applicable amendments, notices, and revisions, shall be as specified in the contract.
2. Format. The cmi5 format that uses JavaScript Object Notation (JSON) will be used. All cmi5 data must be conformant to the requirements identified in the cmi5 specification.
3. Content. The content will utilize the cmi5 specification and will implement the requirements defined in the specification.
4. Log Files. The content of the log files will show evidence of successful communication of the cmi5 statements to a Learning Record Store (LRS) and shall include any error codes. The log files that show evidence of the successful cmi5 statements storage and retrieval can be of any text format as long as the statements in the log file are the original cmi5 JSON value.

End of DID#

Chapter 8

Building and Migrating cmi5 Content

In this chapter:

- cmi5 content best practices and common mistakes to avoid
- Migrating legacy SCORM content
- Using the course templates to build or migrate content

8.1 Purpose

This chapter is intended to provide the best practices for building new cmi5 content and migrating legacy content to cmi5. The guidance in this chapter, when combined with other resources like Chapter 4 Deep Dive into cmi5 Content and the full cmi5 specification, functions as a resource for content administrators, instructional designers, content authors, content testers, content developers, and other content professionals looking to build cmi5 content or migrate legacy SCORM content to cmi5.

8.1.1 How to Use This Chapter

This chapter is designed to be used as a guide for building and migrating cmi5 content. For more in-depth knowledge and to review the cmi5 requirements, visit the full cmi5 specification. The terms “learning activity” or “content” are used throughout the following sections to denote cmi5 content and to encompass all modalities of cmi5 content, including, but not limited to, authoring or content development tools, traditional courseware, videos, simulations, virtual reality, or augmented reality content. Assignable Unit (AU), the launchable piece of cmi5 content, is also used in accordance with the cmi5 specification. To match the terminology used in the cmi5 specification, LMS is used to describe the system that is “launching” the cmi5 content, but the system may or may not be a traditional Learning Management System. MUST and MUST NOT and SHOULD and SHOULD NOT are capitalized when referring to a requirement of the cmi5 specification.

8.1.2 Who This Chapter Is For

This chapter is for content administrators, instructional designers, content authors, content testers, content developers, and other content professionals wanting to build new cmi5 content or migrate legacy SCORM content.

8.2 cmi5 Content Best Practices and Common Mistakes to Avoid

The following best practices and common mistakes will help content creators and developers migrate existing content and build new cmi5 learning activities.

8.2.1 Implementing the SHOULD and SHOULD NOT Assignable Unit (AU) Aspects of the cmi5 Specification

The following SHOULD and SHOULD NOT aspects will need to be considered and supported in learning activities when necessary depending on the learning activity. Failing to include appropriate SHOULD or SHOULD NOT aspects may cause interoperability concerns. Links to the corresponding part of the cmi5 specification[1] are included for reference.

7.1.2 Last Statement Call

Once the AU has determined that the session will end (e.g., by user action, timeout, or some other means) the AU SHOULD issue a "Terminated" statement.

8.1.2 Content Launch Mechanisms - *fetch*

The AU MUST get the *fetch* value from the query string. The AU MUST make an HTTP POST to the *fetch* URL to retrieve the authorization token as defined in Section 8.2. The AU MUST place the authorization token in the authorization headers of all HTTP requests made to the endpoint using xAPI. The AU SHOULD NOT make more than one post to the *fetch* URL.

8.2.1 Authorization Token *Fetch* URL

The AU SHOULD NOT attempt to retrieve the authorization token more than once.

9.5.4.1 AU Statements That Include Duration

"Terminated" statement

The AU MUST include the "duration" property in "Terminated" statements. The AU SHOULD calculate duration for "Terminated" statements as the time difference between the "Initialized" statement and the "Terminated" statement. The AU may use other methods to calculate the duration based on criteria determined by the AU.

“Completed” statement

The AU MUST include the “duration” property in “Completed” statements. The AU SHOULD calculate duration as the time spent by the learner to achieve completion status.

“Passed” statement

The AU MUST include the “duration” property in “Passed” statements. The AU SHOULD calculate duration as the time spent by the learner to attempt and succeed in a judged activity of the AU.

“Failed” statement

The AU MUST include the “duration” property in “Failed” statements. The AU SHOULD calculate duration as the time spent by the learner to attempt and fail in a judged activity of the AU.

9.5.5.1 Extensions - Progress

The AU may set this value in statements to indicate level of completion. The AU SHOULD NOT set a progress value in a “Completed” statement or if it has previously issued a “Completed” statement for the AU in the current registration.

10.2.2 Document Properties - launchMode

The launch mode is determined by the LMS. There are three possible values:

- **Normal -** Indicates to the AU that satisfaction-related data MUST be recorded in the LMS using xAPI Statements.
- **Browse -** Indicates to the AU that satisfaction-related data MUST NOT be recorded in the LMS using xAPI Statements. When Browse mode is used, the AU SHOULD provide a user experience that allows the user to “look around” without judgment.
- **Review -** Indicates to the AU that satisfaction-related data MUST NOT be recorded in the LMS using xAPI Statements. When Review mode is used, the AU SHOULD provide a user experience that allows the user to “revisit/review” already completed material.

10.2.3 Document Properties - launchParameters

The AU SHOULD get the launchParameters value from the State API Document if the launch parameters were defined in the course structure.

10.2.7 Document Properties - entitlementKey

The AU SHOULD use this data in combination with other data provided from the LMS to determine entitlement.

10.2.7.1 Document Properties - courseStructure

The AU SHOULD use this data in combination with other data provided from the LMS to determine entitlement.

10.2.7.2 Document Properties - alternate

The AU SHOULD use this data in combination with other data provided from the LMS to determine entitlement.

The entitlementKey consists of both the courseStructure and alternate document properties. entitlementKey value example:

```
{ "courseStructure": "xyz-123-9999", "alternate": "abc-456-1111" }
```

11.1 xAPI Agent Profile Data Model - languagePreference

The languagePreference MUST be a comma-separated list of RFC 5646 Language Tags as indicated in the xAPI specification. In the list, languages MUST be specified in order of user preference. In the example below, the user's first preference for language is en-US. The user's second preference for language is fr-FR and the third preference is fr-BE.

```
{
  "languagePreference": "en-US,fr-FR,fr-BE",
  ...
}
```

If the AU supports multiple languages, the AU SHOULD display in the language preference order of the user as in the example above. If the AU supported "zh-CN", "fr-BE", and "fr-FR", it SHOULD display in "fr-FR". If the AU does not support multiple languages, or if no languagePreference is specified in the Agent Profile, it may display in its default language.

13.1.1 AU Metadata - entitlementKey

Data used by the AU to determine if the launching LMS is entitled to use the AU. The AU SHOULD use this data in combination with other data provided from the LMS to determine entitlement. Values are defined by the AU content provider.

Sample value:

```
<entitlementKey>
xyz-123-9999
</entitlementKey>
```

8.2.2 cmi5 Content Best Practices[2]

Titles and descriptions

When creating titles and descriptions, the best practice is to use unique titles and descriptions that are relevant to, and descriptive of, the content.

Provide registration ID in all statements

It is strongly recommended to include the registration ID provided by the LMS at launch time in the context of all statements issued by the LMS and AU.

Documenting Activity ID inventory list

It is a best practice to take an inventory of activity information to avoid ID collisions and multiple IDs for the same activity. The inventory list document should include (at a minimum):

- Activity IDs (IRI)
- Activity names
- Activity descriptions
- Activities that are the Object of an xAPI Statement or contextActivities in an xAPI Statement

Use “Progressed” verb for indicating progress during a session

For recording progress during a session, it is recommended to use a “cmi5 allowed” statement with the “Progressed” verb and a progress extension in the result. The AU may set this value in statements to indicate level of completion. The AU SHOULD NOT set a progress value in a “Completed” statement or if it has previously issued a “Completed” statement for the AU in the current registration. “Progress” statements should not be sent for progress values of 100% as that indicates completion. Once the learner reaches 100%, it is recommended that a “cmi5 defined” “Completed” statement be issued instead.

Use of objectives

Objectives are defined for the course structure, but there is no language in the specification concerning their usage in statements. If an AU is using Objectives in statements, then best practice is to add the objective (with the same objective ID provided in the course structure) to the context activities "grouping" property as an activity type of http://adlnet.gov/expapi/activities/objective from the ADL vocabulary.

Use of objectives (in LMS/course structure)

Objectives are defined for the course structure so that the LMS can associate learning objectives to AU and blocks in the course structure.

AU "masteryScore"

If the LMS issues a "masteryScore", the AU should respond in the following ways:

- If the AU has no notion of scoring, it will not issue "Passed" or "Failed" statements.
- If the AU does not have scoring or the learner does not meet the "masteryScore", then the AU MUST NOT issue a "Passed" statement.
- The AU can also refuse to execute because the "masteryScore" issued is inconsistent with the learning design of the AU. In this situation, the AU should inform the learner why it cannot execute.

Always specify a moveOn criterion in the course structure for each AU

When creating a course structure, always specify a moveOn criterion for each AU. Failing to do so will result in a default of "NotApplicable". This can have the following unintended consequences:

- The course will automatically be marked "Satisfied" immediately upon registration of the learner, even before the learner launches a single AU, when failing to specify moveOn criteria or specifying "NotApplicable" as the moveOn criteria for all AUs in a course.
- If at least one AU has a moveOn criteria specified (other than "NotApplicable"), then the course would be "Satisfied" upon satisfaction of those AUs.

Best practice is to explicitly specify the moveOn for each AU in a course structure to ensure there is no confusion over the satisfaction requirements of a course.

In the absence of returnURL, the AU should close the browser window to exit

If the AU is not a mobile application and the LMS does not provide a returnURL, the best practice to exit the AU is to close the window (window.close()) or direct the learner to manually close the browser window.

AU should use a derived activity ID for “cmi.interaction” statements

When the AU issues statements with an activity type of “cmi.interaction” it should use an Activity ID derived from the AU’s Activity ID by appending the following convention:

```
/test/{TestID}/question/{QuestionID}
```

Where:

`{TestID}` is a vendor-defined identifier for the test embedded in AU.

`{QuestionID}` is a vendor-defined identifier for the test question.

Example:

```
https://example.com/au/xyz123/test/997980ef-2089-4685-97ee-6949541a27e5/question/
3f45e67c-f070-4ded-8fd8-82c4069e8526
```

Persistent AU session state

There are certain situations, such as browser window refresh, that may inadvertently cause cmi5 errors related to operations that must occur only once in a session. To prevent such issues, AUs should be designed to use non-volatile storage (local to the AU) to preserve the state of the following operations that have been performed during the session:

- *Fetch* URL token retrieval (which can only be called once in session).
- All “cmi5 defined” statements that can only be sent once in a session.

Test in the cmi5 Content Test Suite (CTS)

All content should be thoroughly tested in the cmi5 CTS. The full course should be tested to ensure that the entire course is conformant to the cmi5 specification.

8.2.3 Common Mistakes to Avoid[3]

Disregarding the returnURL

Mistake: The AU doesn't redirect the browser to the returnURL when exiting.

Consequence: Since the best practice for the LMS is to use the returnURL, the learner will not be able to return to the LMS user interface after exit. This is also a violation of the minimum conformance requirements of cmi5, and the AU is not conformant.

Using the Activity ID as Publisher ID

Mistake: The AU doesn't use the Publisher ID provided in the State API in context activities of the statements it makes, incorrectly using the LMS generated Activity ID instead.

Consequence: The data in statements are incorrectly documented. The AU is not conformant.

Disregarding the "masteryScore"

Mistake: The AU doesn't issue the proper statements when the score is above the "masteryScore". It uses its own internally-specified score threshold to determine mastery instead.

Consequence: The content is incorrectly issuing "Passed" or "Failed" statements. The AU is ignoring the "masteryScore" as provided by the LMS administrator (or course structure as it currently exists). The AU is not conformant.

Automatic satisfaction immediately on course registration

Mistake: A course structure has not included moveOn Criteria for any of its AUs.

Consequence: The course is immediately satisfied on registration (before the learner launches any AUs in the course). While this is not technically a violation of the cmi5 requirements, it is likely a mistake.

Not respecting Browse and Review launch modes

Mistake: Launch mode values (of Browse and Review) provided by the LMS in the State Document are ignored by the AU.

Consequence: The AU incorrectly issues cmi5 statements other than "Initialized" and "Terminated". The AU is not conformant.

8.3 Migrating Legacy SCORM Content to cmi5

This section is for content professionals wanting to migrate legacy content to the more modern cmi5 specification in order to see the benefits that cmi5 plus xAPI data tracking provides. It is important to note that, depending on your existing course library, content migration will likely be an on-going process and does not have to be completed all at one time.

8.3.1 When to Use cmi5

The following subsection includes a few instances where content professionals may want to consider migrating existing legacy content to cmi5.

Learning activities will be launched from an LMS and one or more of the following are desired:

- **Updating existing content.** Needing to revise or update existing content provides the perfect opportunity to migrate to cmi5.
- **Capturing more detailed learning data from existing courseware.** cmi5 is the preferred specification for when program managers, content authors, or instructional designers desire to see more detailed data about what happens inside the course or learning activities beyond what the SCORM standards support.
- **Modernizing training per the Department of Defense Instruction 1322.26.** Federated platforms looking to move to a modern learning and training ecosystem as described in the DoDI 1322.26 can start the content migration process with a few existing legacy courses.
- **When pop up windows are not desirable.** With cmi5, learning activities (content) can be launched in the same window as the LMS.

8.3.2 SCORM to cmi5 Terminology

To help with the migration process, here are some of the key concepts of SCORM and what those concepts are called in the cmi5 specification. Not all functions of cmi5 behave in the same way as the SCORM features.

Figure 8-1: SCORM to cmi5 Terminology

Section	SCORM	cmi5
Packaging / Structure	Activity (SCO) Identifier	Publisher ID
	Aggregation / Cluster	Block
	Content Package	Course Package
	Launch Data	launchParameters
	Manifest	Course Structure
	Mastery Score	“masteryScore”
	Objective [2]	Objective
	Resource HREF	AU URL
	Rollup (partial)	moveOn
	SCO	AU
	Sequencing [2]	(n/a)
Runtime	Student ID	Actor Inverse Functional Identifier (IFI)
Verbs	`LMSGetValue("cmi.core.entry") = "ab-initio" or "resume" GetValue(“cmi.entry”) = “ab-initio” or “resume” [3]`	Launched [4]
	`LMSInitialize(“”) Initialize(“”)`	Initialized
	`LMSSetValue(“cmi.core.lesson_status”, “completed”) SetValue(“cmi.completion_status”, “completed”)`	Completed

Section	SCORM	cmi5
	`LMSSetValue(“cmi.core.lesson_status”, “passed”)` `SetValue(“cmi.success_status”, “passed”)`	Passed
	`LMSSetValue(“cmi.core.lesson_status”, “failed”)` `SetValue(“cmi.success_status”, “failed”)`	Failed
	Manifest Rollup (partial)	Abandoned [4]
	Manifest Rollup (partial)	Waived [4]
	`LMSFinish("")` `Terminate(“”)`	Terminated
	Manifest Rollup (partial)	Satisfied [4]
Other Runtime Values		
(audio level)	`LMSGetValue(“cmi.student_preference.audio”)` `GetValue(“cmi.learner_preference.audio_level”)`	Agent Profile audioPreference

(bookmarking)	`LMSSetValue(“cmi.suspend_data”, “...”)`
	`LMSSetValue(“cmi.core.lesson_location”, “...”)`
	`SetValue(“cmi.suspend_data”, “...”)`
	`SetValue(“cmi.location”, “...”)`
		Activity State (general)

(duration)	`LMSSetValue(“cmi.core.session_time”, “0000:00:01”)` `SetValue(“cmi.session_time”, “PT1S”)`	cmi5 Statement - result. duration = "PT1S"
(interactions)	`LMSSetValue(“cmi.interactions.n.XXX” (several)` `SetValue(“cmi.interactions.n.XXX”) (several)`	(statements, depending on profile chosen)

Section	SCORM	cmi5
(language)	`LMSGetValue("cmi.student_preference.language")` `GetValue("cmi.learner_preference.language")`	Agent Profile languagePreference
(mode (normal, browse, review))	`LMSGetValue("cmi.core.mode")` `GetValue("cmi.mode")`	Activity State LMS. LaunchData, "launchMode"
(progress)	`SetValue("cmi.progress_measure", "0.5") [2]`	cmi5 statement - result extension https://w3id.org/xapi/cmi5/result/extensions/progress = 0.5

(score)	`SetValue("cmi.score.scaled", "0.5");`
	`SetValue("cmi.score.raw", "50");`
	`SetValue("cmi.score.min", "0");`
	`SetValue("cmi.score.max", "100"); [5]`
		cmi5 statement -
		`{"result": {`
		`"scaled": 0.5,`
		`"raw": 50,`
		`"min": 0,`
		`"max": 100`
		`}}`

Note: Depending on the SCORM version originally used to create the course, not all features listed in the SCORM column will apply.

[1] SCORM 1.2 only

[2] SCORM 2004 only

[3] SCORM read-only value

[4] Statement sent by LMS

[5] SCORM 1.2 did not have a scaled score, so only 2004 mappings are shown here

Source: Rustici Software

8.3.3 cmi5 Extensions

As shown above, the cmi5 specification does not address Sequencing in the same way that SCORM did. To address this, the cmi5 Working Group developed two extensions, which are not part of the specification but are compatible. If the following features apply to your use case, it is recommended that they be used to ensure interoperability:

- requires – Defines a specified list of AUs that must have met their moveOn criteria before a given AU is accessible by the learner. More about the requires extension: https://aicc.github.io/CMI-5_Spec_Current/extensions/requires.html
- collateralCredit – Defines a list of blocks and AUs that the LMS MUST issue "Waived" statements for when a given AU is satisfied. More about the collateralCredit extension: https://aicc.github.io/CMI-5_Spec_Current/extensions/collateralcredit.html

8.3.4 cmi5 Example Course Templates

The following section (8.4) goes into detail about the example course templates that were created as part of the cmi5 CATAPULT project. The course templates were designed to help give content professionals a guide for migrating legacy content to the cmi5 specification.

These example templates are designed to help content professionals better understand the technical components of cmi5 and are not intended to be copied in terms of design or content. This collection of content packages was developed to demonstrate various usage patterns for the packaging, launch, and runtime operation of a cmi5 course. This section introduces the course templates and how they might be applied. Course templates and documentation can be found here: https://adlnet.github.io/CATAPULT.

8.4 Using the Course Templates to Migrate or Build Content

8.4.1 Creating the Course Structure

To import learning activities into an LMS, content professionals will need to provide the course structure and the course package. The course template documentation provides an example course structure with various sections highlighted that may be modified by the instructional designer or course developer.

8.4.2 Adding the JS Libraries

The example course templates provide JS files to copy as well as a cmi5 wrapper file to copy that simplifies some of the higher-level cmi5 workflows. The templates assume content professionals will use both files.

8.4.3 The Course Templates

The cmi5 course templates start with a set of basic HTML pages that contain instructional information about geology. Content professionals should focus on the technical components of cmi5 and not on the course content or design. There are six total example course templates available as part of the cmi5 CATAPULT project, which consist of four traditional framed courses and two responsive courses that perform better in mobile environments.

Simple Single AU

This example demonstrates the most basic content package and includes both framed and responsive examples. The course structure contains only a single AU. The AU itself will be treated by the LMS as “Satisfied” if the course submits either the “Completed” or “Passed” verbs by the moveOn=“CompletedOrPassed” setting. Satisfaction occurs upon launching the course successfully.

Additional behaviors found in this package:

- **Bookmarking.** This course template will store a “bookmark” at each navigation event, using the Activity State as described above. On relaunch of the course, the bookmark will be retrieved if available, and the learner presented with the option to return to that location in the course.
- **Video tracking.** This course template has a playable video and generates xAPI Statements per the xAPI Video Profile. The video uses custom controls that allow it to be played exactly once, to model the simplest possible video playing and tracking experience.

“masteryScore”

The “masteryScore” is an optional mechanism to control the course’s behavior relative to determining passed or failed status. If the AU has scoring and the course structure has a “masteryScore”, this value (or an LMS override of the value, see Section 10.2.4 of the cmi5 specification) MUST be used by the AU as an override behavior to determine the passed or failed status of the registration for that AU. This value is included in the LMS Launch Data activity state, which the AU must fetch. Chapter 4 contains more information about “masteryScore”.

This example includes a simple quiz and requires a certain score to be achieved on the quiz to receive a “CompletedAndPassed” set of statements. The AU requires both passed and completed for the LMS to consider it “Satisfied”, by the moveOn=“CompletedAndPassed” setting.

Additional behaviors found in this package:

- **Interactions.** During the quiz, several interactions are captured as xAPI Statements, including simple fill-in and multiple choice interactions.

This course template includes both framed and responsive examples.

Multiple Top-Level AU

This example takes the contents of the “masteryScore” example and spreads them across several AUs, nested at the topmost level of the course structure. The AUs have appropriate moveOn criteria for their execution.

Pre/Post Test

This example demonstrates a course structure of blocks with each block containing a pre-test, content section, and post-test AU (three AUs per block). At its core, cmi5 does not include any prerequisite or dependency behavior, so in a standard cmi5 implementation, this only provides organizational structure and potentially satisfaction measurement for each block.

This package also includes an example of a cmi5 Extension, a way to extend the valid tags and attributes allowed inside an XML course structure. In this case, the template used a `requires` extension, which allows the course author to indicate to the LMS what AUs must have met their moveOn criteria before the block or AU containing the `<require>` tags is accessible to the learner. Since this is an extension and not core to cmi5, LMSs may choose not to implement this behavior, so it’s important to recognize that this approach to pre/post testing will not necessarily perform equivalently on all cmi5-conformant platforms.

Course templates and documentation can be found here: https://adlnet.github.io/CATAPULT.

8.5 Testing in the cmi5 Content Test Suite (CTS)

All created content should be thoroughly tested in the cmi5 Content Test Suite (CTS) to ensure the content is conformant to the cmi5 specification. The full course should be tested to ensure a passing result from the CTS. Once tested, content testers will receive test reports. The following chapter, Chapter 9 Testing in the cmi5 Content Test Suite (CTS), provides more information about the CTS.

Chapter 9

Testing in the cmi5 Content Test Suite (CTS)

In this chapter:

- The importance of testing content for cmi5 conformance
- About the cmi5 Content Test Suite (CTS)
- Information about the user documentation and installation guidance

9.1 Purpose

This chapter is intended to provide an overview of the CTS, including the importance of testing content in a test suite, about the test suite, and where to go for detailed installation and user guidance. The scope of this chapter is limited to the cmi5 specification only and does not address testing requirements for cmi5 systems (see chapter 14 for the cmi5 LMS Test Suite) or other e-learning standards.

9.1.1 How to Use This Chapter

This chapter should be interpreted as an introduction to the CTS developed as part of the cmi5 CATAPULT project and explanation of why testing in the test suite is important.

The terms “learning activity” or “content” are used throughout the following sections to denote cmi5 content and to encompass all modalities of cmi5 content, including, but not limited to, authoring or content development tools, traditional courseware, videos, simulations, virtual reality, or augmented reality content. Assignable Unit (AU), the launchable piece of cmi5 content, is also used in accordance with the cmi5 specification. To match the terminology used in the cmi5 specification, LMS is used to describe the system that is “launching” the cmi5 content, but the system may or may not be a traditional Learning Management System.

9.1.2 Who This Chapter Is For

This chapter is for program managers, content administrators, instructional designers, content authors, content testers, content developers, and other content professionals looking to use the CTS to test and validate legacy content that was migrated to cmi5 or new cmi5 content that was developed in-house. This chapter is also for program managers looking to test and validate procured content.

9.2 The Importance of Testing cmi5 Learning Activities for Conformance

When building or procuring cmi5 learning activities that will be integrated with other systems, such as an LMS, it is highly beneficial for content professionals to test and validate that the learning activity adheres to the specification and will be compatible across various systems. The CTS allows content professionals to test learning activities to ensure cmi5 conformance and plug and play interoperability across cmi5 conformant systems. Not using the CTS to test learning activities could result in interoperability issues when launched from different LMSs.

The CTS is also a valuable tool for content and acquisitions professionals looking to procure cmi5 content. Having the vendor provide all of the applicable test reports before procurement ensures that the learning activity will be compatible with cmi5 conformant LMSs.

9.3 About the cmi5 Content Test Suite (CTS)

The CTS provides a web-based user interface for cmi5 content conformance testing. The CTS may be locally installed or accessed and deployed from a hosted version provided by the Advanced Distributed Learning (ADL) Initiative. From this interface, program managers, content administrators, instructional designers, content authors, content testers, content developers, and other content professionals are able to upload course packages, launch AUs as part of test sessions, view and download test results, and perform other basic course and test management tasks.

The CTS is intended for content professionals who need to verify that their learning activity is conformant with the cmi5 specification. The CTS allows users to import and validate cmi5 course structures, launch AUs, and see a record of the content's behavior during those launch sessions.

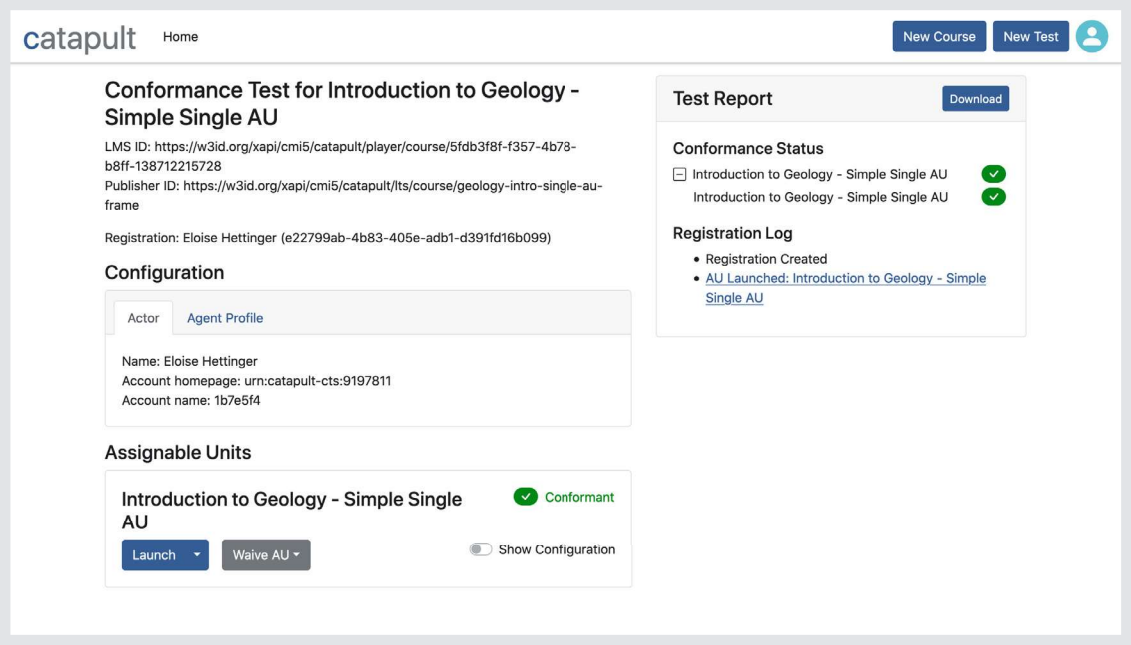
Once a course has been imported, content professionals or testers will be able to see if the learning activity did not meet one or more of the cmi5 requirements. In addition, the user can see the entire specification requirement when they hover over the error message.

9.3.1 User Documentation Information

User documentation was created to help content professionals navigate the CTS tool. The user documentation includes information on how to:

- Sign in for the first time
- Navigate the CTS
- Add a course
- Test a course
- Download the test report

Figure 9-1: Example of a Test Session in the CTS



Source: cmi5 Content Test Suite

The documentation includes the use of a Tutorial to guide users through the entire testing process and a Reference section that goes into more detail about specific parts of the application, including the course list, course detail, and testing.

The CTS user documentation, along with other documentation for the CTS, can be found here: https://adlnet.github.io/CATAPULT

9.3.2 Installation Guidance Information

In order to ensure the installation and deployment of the CTS is executed correctly, installation guidance was created as part of the cmi5 CATAPULT project. The installation guidance includes:

- An overview of the system
- Detailed installation instructions, including required steps and activities
- Detailed configuration instructions, including required steps and activities
- Description of the major tasks involved in the setup, configuration, and deployment of overall resources needed become operational.

Installation and deployment and other CTS documentation can be found here: https://adlnet.github.io/CATAPULT

cmi5 for Systems Professionals

In this section:

- Acquiring cmi5 systems
- cmi5 systems: Acquisition guidance and contracting language examples
- Implementing cmi5 systems
- cmi5 Player Prototype
- cmi5 LMS Test Suite (LTS)

Purpose of this section:

This section is for professionals, such as program managers, systems developers, and acquisitions professionals, who are looking to acquire, implement, or build systems that work with cmi5.

This section answers:

- What are the best practices for acquiring systems that work with cmi5?
- What contracting language should be used when acquiring systems?
- How do I implement cmi5 systems?
- What are the best practices for building cmi5 conformant systems?
- Where can I learn more about the open-source cmi5 Player Prototype?
- Where can I learn more about the cmi5 LMS Test Suite (LTS)?

Chapter 10

Deep Dive Into cmi5 Systems Requirements

In this chapter:

- Integration with an LRS
- Import and validation of course packages
- Registration Creation
- Handling moveOn Criterion
- Launch URL generation
- Sessions
- Handling key cmi5 aspects

10.1 Purpose

This chapter is intended to provide a deeper dive into the systems (LMS) related features of the cmi5 specification. The guidance in this chapter, when combined with other resources like Chapter 13 Building cmi5 Systems, functions as a resource for systems developers and other systems professionals wanting to learn more about the LMS features and requirements as defined in the cmi5 specification. The scope of this chapter is limited to the cmi5 specification only and does not address requirements for systems utilizing SCORM or other standards.

10.1.1 How to Use This Chapter

This chapter is designed to be used as a guide for understanding the most common systems features of cmi5. The term “learning activity” is used to denote cmi5 content and to encompass all modalities of cmi5 content. Assignable Unit (AU), the launchable piece of cmi5 content, is also used in accordance with the cmi5 specification. To match the terminology used in the cmi5 specification, LMS is used to describe the system that is “launching” the cmi5 content, but the system may or may not be a traditional Learning Management System. MUST and MUST NOT and SHOULD and SHOULD NOT are capitalized in accordance with the cmi5 specification.

10.1.2 Who This Chapter Is For

This chapter is for systems developers and other systems professionals wanting to learn more about LMS-related features of the cmi5 specification.

10.2 Integration With a Learning Record Store (LRS)

In order for an LMS to be conformant with the cmi5 specification, the LMS must have an integrated Learning Record Store (LRS). When reviewing the cmi5 specification, it is important to note that the term “LMS” refers to an LMS with an integrated LRS. The LMS MUST conform to all LRS requirements as specified in the xAPI specification.

The LRS is a web service that’s responsible for receiving, storing, and providing access to learning records (xAPI Statements). LRSs validate the Statements to be sure the format of the Statements are conformant to the xAPI requirements. Conformant LRSs will not accept or retain Statements that do not conform to the xAPI specification.

10.3 Import and Validation of Course Packages[1]

In order to be conformant with the cmi5 specification, an LMS MUST provide functionality such that a targeted course package is processed, which results in a course structure import. An LMS MUST support course packages in at least these three file formats:

- Standalone XML file
- 32-bit ZIP Format
- 64-bit ZIP Format

An LMS MAY support alternate course package formats.

The course structure data model states that all leading/trailing whitespace MUST be removed by the LMS on import of the course structure for all of the data elements defined in this section. The following data types are used in the cmi5 course structure data model, see the CourseStructure.xsd (Section 14.0) for specific format:

- **Decimal.** XSD definition: `"xs:decimal"`
- **IRI.** XSD definition: `"xs:anyURI"`
- **String.** XSD definition: `"xs:string"`
- **Langstring.** XSD definition: `<xs:element name="langstring" maxOccurs="unbounded" minOccurs="1"/>`
- **objectiveReference.** XSD definition: `<xs:element name="objectives" type="referencesObjectivesType" minOccurs="0"/>`

10.4 Registration Creation

In cmi5, a registration is an enrollment instance of a learner in a course, and the registration ID uniquely identifies this. The registration ID persists as the learner progresses throughout the learning activity until completion and during review of a completed learning activity. A new registration is created for new enrollment instances, which could be for recurrent courses or retaking courses.

In regards to registration, the LMS MUST place the value for registration in the query string. This value MUST be based on the authenticated learner's corresponding enrollment for the learning activity (course) that the launched AU is a member of.

Within xAPI Statements, the value for the registration property used in the context object MUST be the value provided by the LMS. The LMS MUST generate this value and pass it to the AU via the launch URL.

The LMS MUST evaluate moveOn criteria in the course structure at the time of registration (see next section for handling moveOn criteria). For example, AUs with moveOn criteria of "Not Applicable" in the course structure would be evaluated and could generate "Satisfied" statements at this time.

10.5 Handling moveOn Criteria

The moveOn criteria is used by the LMS to determine if an AU has been sufficiently satisfied for the purposes of determining overall course satisfaction or determining if pre-requisites were met for other activities. moveOn values are as follows:

- **“Passed”:** The LMS considers the AU satisfied when it receives a statement with the verb “Passed”.
- **“Completed”:** The LMS considers the AU satisfied when it receives a statement with the verb “Completed”.
- **“CompletedAndPassed”:** The LMS considers the AU satisfied when it receives statements with the verbs “Completed” and “Passed”.
- **“CompletedOrPassed”:** The LMS considers the AU satisfied when it receives a statement with either of the verbs “Completed” or “Passed”.
- **“NotApplicable”:** The LMS considers the AU satisfied.[2]

moveOn MUST be provided in the LMS Launch Data for the AU plus registration. The LMS MUST add moveOn to the context of a “Launched” statement. The moveOn value written in the State API Document MAY be different than the one in the course structure (e.g. based on administrative rules defined by the LMS).

The default value for moveON criteria is “NotApplicable”. The best practice is to always specify a moveOn criteria for each AU. Failing to do so will result in one of the following unintended consequences:

- The course will automatically be marked “Satisfied” immediately upon registration of the learner, even before the learner launches a single AU, when failing to specify moveOn criteria or specifying “NotApplicable” as the moveOn criteria for all AUs in a course.
- If at least one AU has a moveOn criteria specified (other than “NotApplicable”), then the course would be “Satisfied” upon satisfaction of those AUs.

The best practice is to explicitly specify the moveOn for each AU in a course structure to ensure there is no confusion over the satisfaction requirements of a course.

10.6 Launch URL Generation

The AU MUST be launched by the LMS with a URL having query string launch parameters as defined in this section (Section 8.0 of the cmi5 specification). The launch parameters MUST be name/value pairs in a query string appended to the URL that launches the AU.

If the AU's URL requires a query string for other purposes, then the names MUST NOT collide with named parameters defined below. The AU MUST have the ability to process the query string launch parameters in any order. Each value for the associated names MUST be URL-encoded.

The format of the launching URL is as follows:

```
<URL to AU>
?endpoint=<URL to LMS Listener>
&fetch=<Fetch URL for the Authorization Token>
&actor=<Actor>
&registration=<Registration ID>
&activityId=<AU activity ID>
```

Example:

```
http://www.example.com/LA1/Start.html
?endpoint=http://lrs.example.com/lrslistener/
&fetch=http://lms.example.com/tokenGen.htm?k=2390289x0
&actor={“objectType”: “Agent”, “account”:
{“homePage”: “http://www.example.com”, “name”: “1625378”}}
&registration=760e3480-ba55-4991-94b0-01820dbd23a2
&activityId=http://www.example.com/LA1/001/intro
```

(For readability the above example is not URL encoded and is not intended for use.)

The values for the URL launch parameters and the LMS and AU usage are provided:

Figure 10-1: URL Launch Parameters[3]

endpoint	
Description	A URL to the LMS listener location for xAPI requests to be sent to.
LMS Usage	The LMS MUST place the endpoint in the query string.
AU Usage	The AU MUST get the endpoint value from the query string. The AU MUST use the endpoint value as the base endpoint for xAPI requests.

fetch	
Description	The *fetch* URL is used by the AU to obtain an authorization token created and managed by the LMS. The AU being launched uses the authorization token.
LMS Usage	The LMS MUST place the *fetch* in the launch URL (see Section 10.9 Handling *fetch* URL below for more about the *fetch* URL).
AU Usage	The AU MUST get the *fetch* value from the query string. The AU MUST make an HTTP POST to the *fetch* URL to retrieve the authorization token as defined in Section 8.2 of the cmi5 specification. The AU MUST place the authorization token in the Authorization headers of all HTTP requests made to the endpoint using the xAPI. The AU SHOULD NOT make more than one post to the *fetch* URL.
actor	
Description	A JSON object of objectType “Agent” (as defined in the xAPI specification) that identifies the learner launching the AU so the AU will be able to include it in xAPI requests.
LMS Usage	The LMS MUST populate the actor parameter in the query string based on the authenticated learner's identity. The actor property MUST be defined by the LMS as follows. The actor property for all “cmi5 defined” statements MUST be of objectType “Agent”. The actor property MUST contain an “account” IFI as defined in the xAPI specification. The LMS SHOULD create this parameter with an object that is specific to the LMS instance that does NOT include sensitive PII of the learner.
AU Usage	The AU MUST get the actor value from the query string. The AU MUST use the actor value in xAPI requests that require an “actor” property or that require an “agent” parameter.
registration	
Description	A registration ID corresponding to the learner's enrollment for the AU being launched.
LMS Usage	The LMS MUST place the value for registration in the query string based on the authenticated learner's corresponding enrollment for the course that the AU being launched is a member of.
AU Usage	The AU MUST get the registration value from the query string. The AU MUST use the registration value in xAPI requests that require a registration.

activityID	
Description	The activity ID of the AU being launched.
LMS Usage	The LMS MUST generate a unique activityId for the AU. The LMS MUST place its value in the query string. The activityId generated MUST NOT match the AU's ID (publisher ID) from the course structure (See Section 13.1.4 of the cmi5 specification). The LMS MUST use the same generated activityId on all subsequent launches (for the same AU) within the same registration. The LMS SHOULD use the same generated activityId (for the same AU) for all registrations.
AU Usage	The AU MUST get the activityId value from the query string. The AU MUST use the activityId value as the ID property of the object in all “cmi5 defined” statements.

10.7 LMS.LaunchData[4]

The cmi5 State Data Model is held in an Activity State with a stateId of “LMS.LaunchData”, which is unique per learner registration and AU. This state is essentially the method for passing “runtime configuration” to the AU at launch.

Example of LMS.LaunchData that the AU can Retrieve Through the xAPI Activity State API:

```
{
  “launchMode”: “Normal”,
  “masteryScore”: null,
  “moveOn”: “Passed”,
  “contextTemplate”: {
    “contextActivities”: {
      “grouping”: [
        {
          “id”: “http://id.tincanapi.com/activity/software/cmi5-AU-
Simulator/1.4.0/Nested-Block-Single-AU/Outer-Block/Inner-Block/AU”
        }
      ]
    },
    “extensions”: {
      “https://w3id.org/xapi/cmi5/context/extensions/sessionid”: “e5c1de64-90c8-
4549-a08f-074696876809”
    }
  },
  “entitlementKey”: {},
  “returnURL”: “http://localhost:63399/api/v1/sessions/26/return-url”
}
```

(The code above is for example purposes only and not intended for use.)

An important subset of the LMS.LaunchData document properties are as follows:

Figure 10-2: Key LMS.LaunchData Document Properties

contextTemplate	
Description	A context template for the AU being launched.

LMS Usage	The LMS MUST include a “contextTemplate” object and MUST include the following values:
	• The value for session ID placed in an “extensions” property with the ID is defined as follows:
	  ◦ The value for session ID MUST be generated by the LMS.
	  ◦ The LMS MUST record the session ID in the State API prior to launching an AU.
	  ◦ The LMS MUST provide the session ID in the context as an extension for all “cmi5 defined” and “cmi5 allowed” statements it makes directly in the LRS.
	• The publisher ID Activity is defined as follows:
	  ◦ Prior to launching an AU, the LMS MUST include an Activity object with an “ID” property whose value is the unaltered value of the AU’s “ID” attribute from the course structure (see Section 13.1.4 of the cmi5 specification). This “ID” attribute may be found in the “grouping” context activities list in the contextTemplate as described in the State API.
	  ◦ The LMS MUST include the publisher ID Activity in the “grouping” context activities list for all “cmi5 defined” and “cmi5 allowed” statements it makes directly in the LRS

AU Usage	The AU MUST get the contextTemplate value from the LMS.LaunchData State document. The AU MUST NOT modify or delete the LMS.LaunchData State document. The AU MUST use the contextTemplate as a template for the context property in all xAPI Statements it sends to the LMS. While the AU may include additional values in the Context object of such Statements, it MUST NOT overwrite any values provided in the contextTemplate. NOTE: this will include the session ID specified by the LMS.

launchMode	
Description	The launch mode is determined by the LMS. There are three possible values: • **Normal -** Indicates to the AU that satisfaction-related data MUST be recorded in the LMS using xAPI Statements. • **Browse -** Indicates to the AU that satisfaction-related data MUST NOT be recorded in the LMS using xAPI Statements. When Browse mode is used, the AU SHOULD provide a user experience that allows the user to "look around" without judgment. • **Review -** Indicates to the AU that satisfaction-related data MUST NOT be recorded in the LMS using xAPI Statements. When Review mode is used, the AU SHOULD provide a user experience that allows the user to "revisit/review" already completed material.
LMS Usage	The LMS MUST include a value for launchMode.
AU Usage	The AU MUST conform to the following based on the value of launchMode: • **Normal -** The AU MUST send "Initialized" and "Terminated" verb statements. The AU MAY send other "cmi5 defined" statements per the requirements defined in Section 9.3 of the cmi5 specification. • **Browse -** The AU MUST send "Initialized" and "Terminated" verb statements. The AU MUST NOT send other "cmi5 defined" statements. • **Review -** IThe AU MUST send "Initialized" and "Terminated" verb statements. The AU MUST NOT send other "cmi5 defined" statements.

See the full property list in Section 10.2 of the cmi5 specification.

10.8 Sessions

10.8.1 Session Creation and Tracking

The session is a period of time that is marked by the launch of an AU and continues until its termination or abandonment. The session must have a session ID, which is a unique identifier (string) for a single AU launch session based on actor and course registration. The LMS MUST generate the value for the session ID. The LMS MUST record the session ID in the State API prior to launching an AU, as part of the LMS.LaunchData.

10.8.2 Session and Statement Validation

In the LMS.LaunchData Activity State, the LMS MUST include a contextTemplate object that includes the value for session ID placed in an “extensions” property with the ID. In conjunction with the LMS.LaunchData example in the previous section, the relevant subsection of the JSON document is:

```
{
  "extensions": {
    "https://w3id.org/xapi/cmi5/context/extensions/sessionid": "e5c1de64-
90c8-4549-a08f-074696876809"
  }
}
```

The LMS MUST provide the session ID in the context as an extension for all “cmi5 defined” and “cmi5 allowed” statements it makes directly in the LRS.

10.9 How the LMS Handles Key Aspects of the cmi5 Specification

10.9.1 *fetch* URL

The *fetch* URL is used by the AU to obtain an authorization token created and managed by the LMS. The LMS MUST place the *fetch* in the launch URL. The authorization token is used by the AU being launched. The authorization token returned by the *fetch* URL MUST be limited to the duration of a specific user session.

The AU SHOULD NOT attempt to retrieve the authorization token more than once. The *fetch* URL is a “one-time use” URL and subsequent uses SHOULD generate one of the following errors:

- **Already in Use or Expired.** Token has been used before or the AU session has expired.
- **General Security Error.** All other security issues including invalid tokens.
- **General Application Error.** Application or environment failures.

The LMS MUST include the *fetch* name/value pair in the launch URL. The AU MUST make an HTTP POST to the *fetch* URL to retrieve an authorization token. Note that an HTTP GET is not allowed in order to prevent caching of the request.

The *fetch* URL MUST return a JSON structure using a Content-Type of "`application/json`". The structure MUST be an object with the property "`auth-token`" in the first successful response. An example JSON structure is shown below:

```
{
  "auth-token": "QWxhZGRpbjpvcGVuIHNlc2FtZQ=="
}
```

The AU MUST place the "auth-token" in the HTTP header, as defined in RFC 1945 - 11.1 Basic Authentication Scheme, in all subsequent xAPI communications with the LMS. The authorization token returned by the *fetch* URL MUST be limited to the duration of the session.[5]

10.9.2 Handling returnURL

If the LMS requires the AU (in a web-browser environment) to redirect the learner when he or she exits the AU, then the returnURL is used by the LMS when launching the AU. The LMS MAY include the returnURL when the learner SHOULD be redirected to the returnURL on exiting the AU.[6]

10.9.3 Handling the "Abandoned" Verb

The "Abandoned" verb indicates that the AU session was abnormally terminated by either a learner's action or due to a system failure. If an LMS launches an AU with the same registration as an active session, the previous session with that registration MUST be abandoned. The LMS MUST record an "Abandoned" statement on behalf of the AU indicating an abnormal session termination. The LMS MUST NOT allow any statements to be recorded for a session after recording an "Abandoned" statement.[7]

10.9.4 Handling the “Waived” Verb

The “Waived” verb is used to indicate that the LMS has determined that the AU requirements were met by means other than the moveOn criteria and may be skipped by the learner. In this instance, the LMS MUST use the “Waived” verb in a statement recorded in the LRS when it determines that the AU may be waived. A statement containing a “Waived” verb MUST include a “reason” in the extension property of the statement result. The value SHOULD be one of the following:

- **Tested Out.** A learner completed an assessment to waive the AU.
- **Equivalent AU.** The learner successfully completed an equivalent AU (in the same course) to waive the AU.
- **Equivalent Outside Activity.** The learner successfully completed an equivalent activity outside of the course to waive the AU.
- **Administrative.** The LMS administrative user marked the AU complete.

The LMS MUST generate a unique session ID for the statement containing a “Waived” verb and MUST NOT issue any other statements (except for statements with the “Satisfied” verb) using that session ID. The LMS MUST NOT issue multiple statements with “Waived” for the same AU within a course registration.[8]

Chapter 11

Acquiring cmi5 Systems

In this chapter:

- When to acquire cmi5 conformant systems
- Best practices for acquiring cmi5 conformant systems
- What to ask vendors

11.1 Purpose

This chapter is intended to provide acquisition guidance and best practices for acquiring systems that support the cmi5 specification. This chapter, in conjunction with Chapter 12 cmi5 Systems Acquisition Guidance and Contracting Language Examples, functions as a resource for new systems acquisitions that require cmi5. The guidance in this chapter, when combined with other sources, should provide sufficient cmi5 recommendations for acquisition professionals, program managers, and similar roles to include in new cmi5 systems procurement contracts. The scope of this chapter is limited to the cmi5 specification only and does not address requirements for content utilizing SCORM or other standards. This chapter presents a range of possible considerations for acquiring systems, and as the standard evolves and progresses, the considerations may change over time.

11.1.1 How to Use This Chapter

This section should be interpreted as the authoritative guide for acquiring cmi5 conformant systems. This chapter covers the cmi5 systems acquisition requirements and includes information on when to acquire a cmi5 conformant system, best practices, and what to ask vendors.

The terms “learning activity” or “content” are used throughout the following sections to denote cmi5 content and to encompass all modalities of cmi5 content. Assignable Unit (AU), the launchable piece of cmi5 content, is also used in accordance with the cmi5 specification. To match the terminology used in the cmi5 specification, LMS is used to describe the system that is “launching” the cmi5 content, but the system may or may not be a traditional Learning Management System.

11.2 When to Acquire cmi5 Conformant Systems

This chapter specifies only cmi5 contracting requirements for cmi5 systems, but it’s important for acquisition professionals to understand the use cases for acquiring cmi5 content as well as acquiring a cmi5 conformant system.

Acquire cmi5 content if learning activities will be launched from an LMS and one or more of the following are true:

- **Following Option 2 in the DoDI 1322.26 fungible references.**[1] DoDI 1322.26 states that DoD Components shall begin migrating toward the use of xAPI standards with acquiring and maintaining a cmi5 conformant LMS and an xAPI LRS as one of the xAPI implementation paths.
- **Incorporating xAPI enabled activities.** cmi5 is the preferred xAPI Profile when incorporating xAPI enabled activities that will be launched from an LMS. Using an xAPI Profile other than cmi5, such as the Launch Profile or SCORM Profile, could cause problems with data portability and semantic interoperability.
- **Implementing new training modalities.** Legacy SCORM standards are not extensible enough to track newer training modalities, such as simulations, virtual reality, augmented reality, and non-browser based applications. cmi5 is the preferred specification to use when incorporating these newer modalities into training programs and launching from an LMS.
- **Acquiring learning activities that are hosted outside of an LMS.** Unlike SCORM, cmi5 supports Content as a Service (CaaS) models of delivery so that content can be stored outside of an LMS.

cmi5 content must be launched from a cmi5 conformant LMS, so acquiring a cmi5 conformant system is a necessary step when incorporating cmi5 content.

11.3 Best Practices for Acquiring cmi5 Systems

The following content best practices should be considered when acquiring cmi5 systems. Example contracting language is provided in the next chapter of this best practices guide.

11.3.1 Passed the cmi5 LMS Test Suite (LTS)

All cmi5 conformant systems should be tested in the cmi5 LMS Test Suite (LTS). Once tested, the vendor will need to provide the JUnit XML file to prove that the system is conformant to the cmi5 specification. If the system has not been tested or has not passed the LTS, it should be considered non-conformant to the cmi5 specification. In addition, the vendor should perform regular testing of the system with each new release and any time the cmi5 specification is updated, in the LTS. This should be done, either through an automated script or manually, to ensure continued conformance to the specification. Any time the cmi5 specification is updated, the vendor should provide evidence that the system has been tested and conforms to the latest version of the specification.

11.3.2 Support the SHOULD and SHOULD NOT Aspects of the cmi5 Specification

The LMS will need to support all of the LMS specific SHOULD and SHOULD NOT aspects of the cmi5 specification. Failing to incorporate one of the SHOULD or SHOULD NOT aspects may cause interoperability issues. Each aspect includes a link to the corresponding parts of the cmi5 specification[2] for reference.

6.1 LMS Course Handling Requirements

- The LMS SHOULD implement a means to create, edit, and maintain course structures.
- The LMS SHOULD implement the export of the course data structure defined in Section 13 and the Course Package defined in Section 14.
- The LMS SHOULD provide a user interface to the LMS administrative users to create and edit course structures internally.

6.3 LMS Statement API Requirements

The LMS SHOULD reject statements that conflict with the “Statement API” requirements as defined in Section 9.

8.1.3 Content Launch Mechanisms - actor

The LMS MUST populate the actor parameter in the query string based on the authenticated learner’s identity conforming to Section 9.2. The LMS SHOULD create this parameter with an object that is specific to the LMS instance that does NOT include sensitive PII of the learner.

8.1.5 Content Launch Mechanisms - activityId

The LMS MUST generate a unique activityId for the AU. The LMS MUST place its value in the query string. The activityId generated MUST NOT match the AU’s ID (publisher ID) from the course structure (see Section 13.1.4 - AU Metadata). The LMS MUST use the same generated activityId on all subsequent launches (for the same AU) within the same registration. The LMS SHOULD use the same generated activityId (for the same AU) for all registrations.

8.2.1 Authorization Token *fetch* URL - Duplicate call to *fetch* URL

The *fetch* URL is a “one-time use” URL and subsequent uses SHOULD generate an error (see Section 8.2.3).

9.3.9 “Satisfied” Verb

The LMS SHOULD NOT issue multiple statements with “Satisfied” for the same block or course within a course registration for a given learner.

9.5.4.2 LMS statements that include duration: “Abandoned” statement

The duration property MUST be included in “Abandoned” statements. The LMS SHOULD use LMS specific methods (if available) to determine the duration if it has more accurate means of session time calculation than timestamp differences between statements. In the absence of such methods, the duration property MUST be set as the total session time, calculated as the time between the “Launched” statement and the last statement (of any kind) issued by the AU.

11.3.3 General cmi5 Systems Acquisition Best Practices

The following are suggested best practices that LMS vendors should be following and that systems professionals wanting to acquire cmi5 conformant systems should consider during the procurement process.

Use of Objectives (in LMS/Course Structure)

If an LMS has features for managing or reporting on objectives, it uses the objectives defined in the course structure (to make the association of objectives to the AUs and blocks).

LMS always implements the “returnURL”

LMS should always implement the “returnURL”. The LMS does not spawn a new window to launch an AU (i.e. “popup”). If the “returnURL” is not implemented, then the learner cannot return to the LMS user interface after exit.

Fetch URLs

The LMS handles *fetch* URLS in the following ways:

- The *fetch* URL is unique for each session.
- The *fetch* URL only returns an authorization token on the first call.
- The *fetch* URL does not reuse authorization tokens.
- The *fetch* URL returns an error if an HTTP method other than POST is used.
- The authorization token is stored in non-volatile storage.

LMS "masteryScore"

The LMS uses caution when adding a "masteryScore" to the AU course structure entry if they are not present in the original course structure, as some AUs may not be designed to handle scores.

Launching applications on mobile devices

The LMS uses one of the following options to launch AUs that are applications on mobile devices:

Option 1: Uses an application protocol in the launch URL.

Option 2: Uses an HTML wrapper to launch the application. The AU is an HTML page that directs from the mobile browser to the application.

LMS error handling of non-conforming cmi5 statements

The LMS rejects statements from the AU that don't conform to the cmi5 specification and returns an HTTP error.

LMS creates "Satisfied" statements for AUs

The LMS creates a "cmi5 allowed" statement with a "Satisfied" verb when an AU has met its moveOn criteria. The statement also includes the same AU activityId used in "cmi5 defined" statements.

LMS should use the session ID and authentication token to validate actor

When the LMS receives a statement, it verifies that the actor in the statement matches the actor provided on the launch URL and that the authentication token provided was the same one issued for that specific launch session. If these do not match, then the LMS/LRS rejects the statement with an HTTP error.[3]

11.4 What to Ask Vendors

The following section is designed to help acquire cmi5 conformant systems by providing questions for the vendor to answer in a Request for Proposal (RFP), Data Item Description (DID) deliverable, Contract Data Requirements List (CDRL), or other procurement contract.

11.4.1 LMS and Content Launching Systems Providers

Conformance to the cmi5 specification.

- Does the LMS conform to the cmi5 specification version Quartz or the latest version of the cmi5 specification?
- Does the LMS follow all of the MUST and MUST NOT aspects of the cmi5 specification version Quartz or later?
- Does the LMS allow for all of the LMS applicable SHOULD and SHOULD NOT aspects of the cmi5 specification version Quartz or later?
- Is the system regularly tested in the cmi5 LMS Test Suite, either using an automated script or manually, to ensure continued conformance to the cmi5 specification version Quartz or the latest version of the cmi5 specification? Testing should occur anytime the cmi5 specification is updated and on a regular basis, such as annually.

Testing in the cmi5 LMS Test Suite (LTS).

- Has the LMS been tested in the cmi5 LMS Test Suite (LTS)?
- During testing, were all of the steps of the procedure document followed? Did all steps produce the correct results to pass the cmi5 LMS Test Suite (LTS)?
- Will you provide the JUnit XML file?

Following the cmi5 best practices for LMSs.

- If the LMS has features for managing or reporting on objectives, does it use the objectives defined in the course structure?
- Does the LMS always implement the “returnURL”?
- Does the LMS handle *fetch* URLs in accordance with the LMS best practices for *fetch* URLs?
- When launching applications on mobile devices, does the LMS use either an application protocol in the launch URL or an HTML wrapper to launch the application?

- Does the LMS reject statements from the AU that don't conform to the cmi5 specification and return an error?
- Does the LMS create a "cmi5 allowed" statement (with a "Satisfied" verb) when an AU has met its moveOn criteria?
- When the LMS receives a statement, does it verify that the actor in the statement matches the actor provided on the launch URL and that the authentication token provided was the same one issued for that specific launch session?

Chapter 12

cmi5 Systems Acquisition Guidance and Contracting Language Examples

In this chapter:

- Acquisition guidance and contracting language examples
- cmi5 systems contracting language examples
- Contracting pitfalls to avoid

12.1 Overview

This section is intended to provide contracting guidance for acquisition professionals to use as a resource when procuring LMSs and systems that need to support cmi5 and includes cmi5 contracting requirements and contracting language examples. The example language in this document can be reused and augmented with additional contractual requirements as necessary. The scope of this section is limited to only cmi5 and does not address contracting requirements for systems conforming to SCORM, xAPI, or other standards.

Department of Defense Instruction (DoDI) 1322.26 fungible references lists cmi5 (Option 2) as one of the standards and specifications that DoD organizations should use when acquiring distributed learning solutions. The following contracting language and examples will help acquisition professionals looking to acquire systems conforming to SCORM or other standards.

12.1.1. How to Read This Section

This section should be interpreted as the authoritative contracting requirements for cmi5 conformant Learning Management Systems (LMSs) or systems.

There are two levels of obligation with regards to cmi5 conformance requirements identified by the terms herein. Since these cmi5 requirements are intended to be reused as part of the contract requirements in a Request for Proposals (RFP), Requests for Information, Sources Sought Notifications, Statements of Work, or other types of procurement documents, “MUST” or “MUST NOT” and “SHOULD” or “SHOULD NOT” are only used when referring to the cmi5 specification and “SHALL” or “SHALL NOT” are used as part of the contract language. This is done intentionally in order to avoid confusion between what is required as part of the cmi5 specification and what is required of the contractor. Not following these recommendations could risk interoperability and and/or lead to cmi5 deliverables not being accepted.

1. **SHALL OR SHALL NOT.** If a product fails to implement a SHALL or SHALL NOT requirement of the contract and/or fails to implement a “MUST” or “MUST NOT” requirement of the cmi5 specification, the product is considered non-conformant to the requirements.
2. **WILL.** Used to indicate the Government will complete a task.

The terms “learning activity” or “content” are used throughout the contracting language to denote cmi5 content and to encompass all modalities of cmi5 content, including but not limited to traditional courseware, videos, simulations, virtual reality, or augmented reality content. Assignable Unit (AU), the launchable piece of cmi5 content, is also used in the contracting in accordance with the cmi5 specification. To match the terminology used in the cmi5 specification, LMS is used to describe the system that is “launching” the cmi5 content, but the system may or may not be a traditional Learning Management System.

12.2 cmi5 Systems Contracting Language Examples

12.2.1 General cmi5 Requirements and Example cmi5 Contracting Language

The following general requirements and example contracting language should be included in all acquisitions for systems that are cmi5 conformant. This language provides a foundation for more detailed requirements. The examples below can be used without modification, but it is expected that updates may be required based on specific requirements of the cmi5 conformant system. Additionally, adherence to the cmi5 specification should be referenced throughout the acquisition process for any type of training and education system. These include Requests for Information, Sources Sought Notifications, Statements of Work, and other types of requests for solutions.

Requirement	Contracting Language Example
cmi5 Specification Conformance	The Contractor SHALL develop the system such that it is conformant to the cmi5 Specification version Quartz (or latest version) by adhering to the cmi5 specification and being validated in the cmi5 LMS Test Suite. The Contractor SHALL regularly test the LMS with each new release to ensure that the LMS continues to be conformant to the cmi5 specification. The Contractor SHALL regularly test the LMS with each new release to ensure that the LMS continues to be conformant to the cmi5 specification.
Record of cmi5 Specification Conformance	The Contractor SHALL provide verification that the system has been tested in the cmi5 LMS Test Suite and provide the JUnit XML file.

Requirement	Contracting Language Example
LMS Course Handling Requirements	The Contractor SHALL develop the cmi5 conformant LMS in such a way that: • The LMS SHOULD implement a means to create, edit, and maintain course structures. • The LMS SHOULD implement the export of the course data structure defined in Section 13. • The LMS SHOULD implement the export of the Course Package as a standalone XML file, 32-bit Zip format, or a 64-bit Zip format as defined in Section 14 of the cmi5 specification. • The LMS SHOULD provide a user interface to the LMS administrative users to create and edit course structures internally.
LMS Statement API Requirements	The Contractor SHALL develop the system such that the LMS SHOULD reject statements that conflict with the "Statement API" requirements as defined in Section 9 of the cmi5 specification.
Content Launch Mechanisms - activityId	The Contractor SHALL develop the system such that the LMS SHOULD use the same generated activityId (for the same AU) for all registrations.

Authorization Token *fetch* URL - Duplicate call to *fetch* URL	The Contractor SHALL develop the system such that the *fetch* URL is a "one-time use" URL and MUST NOT return an "auth-token" more than once. Subsequent requests made to the *fetch* URL during the session SHOULD generate an error. An example JSON structure is shown below:
	`{`
	`"error-code": "1",`
	`"error-text": "The authorization token has already been returned."`
	`}`
	The following error-code values are allowed:
	Code 1: Already in Use or Expired - Token has been used before or the AU session has expired
	Code 2: General Security Error - All other security issues including invalid tokens
	Code 3: General Application Error - Application or environment failures
	The values for error-text are defined by the LMS.

cmi5 "Satisfied" Verb	The Contractor SHALL develop the system such that the LMS SHOULD NOT issue multiple statements with "Satisfied" for the same block or course within a course registration for a given learner.

12.2.2 cmi5 Requirements for LMSs and Example cmi5 Contracting Language Necessary for Specific Uses

The following requirements and sample contracting language should be included in all cmi5 conformant LMS acquisitions. The samples below can be used without modification, but it is expected that updates, modifications, and additions may be required.

Requirement	Contracting Language Example
Content Launch Mechanisms - actor	The Contractor SHALL develop the system such that the LMS MUST populate the actor parameter in the query string based on the authenticated learner's identity and conforms to the following: The Actor property MUST be defined by the LMS. The Actor property for all "cmi5 defined" statements MUST be of objectType "Agent". The Actor property MUST contain an "account" IFI as defined in the xAPI specification. The LMS SHOULD create this parameter with an object that is specific to the LMS instance that does NOT include sensitive PII of the learner.
LMS Statements That Include Duration - "Abandoned" Statement	The Contractor SHALL develop the system such that the duration property MUST be included in "Abandoned" statements. The LMS SHOULD use LMS specific methods (if available) to determine the duration if it has more accurate means of session time calculation than timestamp differences between statements.
Implement the "returnURL"	The Contractor SHALL develop the system such that it follows the best practice that the LMS does not spawn a new window to launch an AU (i.e. "popup"). Depending on the settings it could take the following actions to launch an AU: • "OwnWindow" – Redirect same window to AU location • "AnyWindow" – Either redirect in the same window or use iframe, LightBox, etc.
Launching Application(s) on Mobile Devices	One of the following options should be used to launch AUs that are application(s) on mobile devices: **Option 1:** Use an application protocol in the launch URL. • AU is an application. • AU has a URL with a protocol LMS launches Application using URL with application protocol. • An application redirecting to a browser is not useful. If using application protocol to launch, don't use "returnURL". **Option 2:** Use an HTML wrapper to launch the application. AU is an HTML page that directs from the mobile browser to the application.

12.3 Contracting Pitfalls to Avoid

The following general contracting pitfalls should be considered in all cmi5 conformant LMS (systems) acquisitions. This language provides a foundation for areas to be aware of when acquiring systems, but contracting pitfalls are not limited to only these areas. It is expected that updates may be required to the below points based on specific requirements and as cmi5 conformant systems evolve.

Not specifying that the LMS either has an integrated Learning Record Store (LRS) or will work with the buyer's preferred LRS.

An LRS is a key component of cmi5 and xAPI. The contracted LMS must have either an integrated LRS or be compatible with the preferred LRS of the federated platform that is acquiring the LMS.

Failing to implement the SHOULD and SHOULD NOT aspects of the specification correctly.

The LMS will need to implement the LMS SHOULD or SHOULD NOT aspects of the cmi5 specification as instructed in the specification in order to maintain conformance. The cmi5 LMS Test Suite does not currently test for SHOULD and SHOULD NOT aspects of the cmi5 specification, but it is a best practice to support all of the SHOULD and SHOULD NOT aspects of the cmi5 specification.

Failing to consider the LMS cmi5 best practices.

The LMS vendor will need to be aware of, and support, the cmi5 LMS best practices that are defined in the contracting language as well as applicable ones not specifically listed. If there are specific best practices that must be implemented, those best practices should be included in the contracting language.

Chapter 13

Building cmi5 Systems

In this chapter:

- cmi5 systems best practices
- Using the cmi5 Player Prototype to build a system
- Testing in the cmi5 LMS Test Suite (LTS)

13.1 Purpose

This chapter is intended to provide best practices and recommendations for building cmi5 conformant systems. The guidance in this chapter, when combined with other resources like Chapter 10 Deep Dive into cmi5 Systems, Chapter 14 Testing in the cmi5 LMS Test Suite (LTS), and the full cmi5 specification, functions as a resource for systems developers and other systems professionals starting to plan and build cmi5 systems.

13.1.1 How to Use This Chapter

This chapter is designed to be used as a guide for starting to plan and build cmi5 conformant systems. For more in-depth knowledge and to review the cmi5 requirements, visit the full cmi5 specification. Assignable Unit (AU), the launchable piece of cmi5 content, is used throughout this chapter in accordance with the cmi5 specification. To match the terminology used in the cmi5 specification, LMS and system are both used to describe the system that is "launching" the cmi5 content, but the system may or may not be a traditional Learning Management System. MUST and MUST NOT and SHOULD and SHOULD NOT are capitalized in accordance with the cmi5 specification.

13.1.2 Who This Chapter Is For

This chapter is for systems developers, LMS administrators, and other systems professionals wanting to learn the best practices and recommendations for building new cmi5 conformant systems.

The following best practices and recommendations will help systems developers, LMS administrators, and other systems professionals plan and build new conformant cmi5 systems.

13.2 cmi5 Systems Best Practices

13.2.1 Implementing the LMS-Related SHOULD and SHOULD NOT Aspects of the cmi5 Specification

The following SHOULD and SHOULD NOT aspects will need to be integrated into the system. Failing to implement the SHOULD or SHOULD NOT aspects may cause interoperability concerns. Links to the corresponding part of the cmi5 specification[1] are included for reference.

6.1 LMS Course Handling Requirements

- The LMS SHOULD implement a means to create, edit, and maintain course structures.
- The LMS SHOULD implement the export of the course data structure defined in Section 13 and the Course Package defined in Section 14.
- The LMS SHOULD provide a user interface to the LMS administrative users to create and edit course structures internally.

6.3 LMS Statement API Requirements

The LMS SHOULD reject statements that conflict with the "Statement API" requirements as defined in Section 9.

8.1.3 Content Launch Mechanisms - actor

The LMS MUST populate the actor parameter in the query string based on the authenticated learner's identity conforming to Section 9.2. The LMS SHOULD create this parameter with an object that is specific to the LMS instance that does NOT include sensitive PII (Personal Identifiable Information) of the learner.

8.1.5 Content Launch Mechanisms - activityId

The LMS MUST generate a unique activityId for the AU. The LMS MUST place its value in the query string. The activityId generated MUST NOT match the AUs ID (publisher ID) from the course structure (see Section 13.1.4 - AU Metadata). The LMS MUST use the same generated activityId on all subsequent launches (for the same AU) within the same registration. The LMS SHOULD use the same generated activityId (for the same AU) for all registrations.

8.2.1 Authorization Token *fetch* URL

The *fetch* URL is a "one-time use" URL and subsequent uses SHOULD generate an error (see Section 8.2.3).

9.3 Verbs

LMS verb ordering rules are as follows:

- LMS may issue multiple “Satisfied” statements in a session.
- The LMS SHOULD NOT issue multiple statements with “Satisfied” for the same block or course within a course registration for a given learner.
- LMS MUST NOT issue more than one “Abandoned” statement for a session.
- LMS MUST NOT issue more than one “Waived” statement per session and MUST not issue more than one “Waived” statement per registration per AU.

9.3.9 Satisfied

The LMS SHOULD NOT issue multiple statements with “Satisfied” for the same block or course within a course registration for a given learner.

9.5.4.2 LMS Statements That Include Duration

“Abandoned” statement

The duration property MUST be included in “Abandoned” statements. The LMS SHOULD use LMS specific methods (if available) to determine the duration if it has more accurate means of session time calculation than timestamp differences between statements. In the absence of such methods, the duration property MUST be set as the total session time, calculated as the time between the “Launched” statement and the last statement (of any kind) issued by the AU.

13.2.2 cmi5 LMS Best Practices[2]

Use of objectives (in LMS/Course Structure)

Objectives are defined for the course structure so that the LMS can associate learning objectives to AU and blocks in the course structure. If an LMS has features for managing or reporting on objectives, it should use the objectives defined in the course structure (to make the association of objectives to the AUs and blocks).

LMS should always implement the “returnURL”

LMS should not spawn a new window to launch an AU (i.e. “popup”). Depending on the settings, it could take the following actions to launch an AU:

- “OwnWindow” – Redirect same window to the AU location.
- “AnyWindow” – Either redirect in the same window or use an iframe, LightBox, etc

fetch URLs

- The *fetch* URL must be unique for each session.
- The *fetch* URL must only return an authorization token on the first call. Subsequent calls must return an error. It must be a “one-time use” URL.
- The *fetch* URL must not reuse authorization tokens.
- The *fetch* URL should return an HTTP error if an HTTP method other than POST is used.
- Since the *fetch* URL can only be called once, the authorization token should be stored in non-volatile storage.

LMS “masteryScore”

An LMS should use caution when adding “masteryScore” to AU course structure entries if they are not present in the original course structure as the AU may not be designed to handle scores. It is recommended that such changes be tested prior to enrolling learners.

Launching applications on mobile devices

One of the following options should be used to launch AUs that are applications on mobile devices:

- **Option 1:** Use an application protocol in the launch URL.
  - AU is an application.
  - AU has a URL with a protocol. LMS launches the application using a URL with an application protocol.
  - An application redirecting to a browser is not useful. If using an application protocol to launch, don’t use “returnURL”.
- **Option 2:** Use an HTML wrapper to launch the application. AU is an HTML page (wrapper) that directs from the mobile browser to the application.

LMS error handling of non-conforming cmi5 statements

If an AU sends a statement that doesn’t conform to cmi5, the LMS should reject it (see Section 6.3) and return an HTTP error 403.

LMS administrators should use caution when changing the moveOn property

The AU may not have the capability of sending statements required by any moveOn value that was not in the original course structure. It's recommended that all changes to course structures are tested after importing prior to learner assignment.

LMS creates "Satisfied" statements for AUs

The LMS should create a "cmi5 allowed" statement (with a "Satisfied" verb) when an AU has met its moveOn criteria. The statement should also include the same AU activityId used in "cmi5 defined" statements.

LMS should use the session ID and authentication token to validate Actor

When the LMS receives a statement, it should verify that the Actor in the statement matches the actor provided on the launch URL and that the authentication token provided was the same one issued for that specific launch session. If the session ID, authentication token, Actor in statement, and actor do not match, then the LMS/LRS should reject the statement with an HTTP 403 Error.

13.3 Using the cmi5 Player Prototype to Build a System

The open-source cmi5 Player Prototype is freely available and functions as an authenticated, web-based system that connects to an LRS (Learning Record Store). The Player Prototype is intended for systems developers to use for building a minimally viable cmi5 implementation to test and demonstrate cmi5 learning content.

The Player Prototype has the ability to import content from a single list of files and sequences that content dynamically based on the requirements of the cmi5 specification. The Player Prototype was designed for achieving DoD Impact Level 4 (IL4) accreditation.

Developers wishing to implement the cmi5 Player Prototype can use the user documentation and implementation instructions that are available alongside the software in GitHub. The documentation walks users through how to:

- Create tenants
- Get authorization tokens
- Manage courses - import, see details, delete
- Launch AUs
- Manage registrations - create, see details, delete
- Adjust learner preferences
- Waive AUs
- Abandon sessions

The cmi5 CATAPULT Player Prototype documentation can be found here: https://adlnet.github.io/CATAPULT

13.4 Testing in the cmi5 LMS Test Suite (LTS)

Once a cmi5 conformant system has been built, the system should be tested in the cmi5 LMS Test Suite (LTS) to ensure the system is conformant to the cmi5 specification. In addition, the LMS should be tested regularly in the LTS, including with each new release and any time the cmi5 specification is updated. This should be done, either through an automated script or manually, to ensure continued conformance to the specification. Any time the cmi5 specification is updated, the system should be tested in order to ensure that it conforms to the latest version of the specification. The following chapter, Chapter 14 Testing in the cmi5 LMS Test Suite (LTS), provides more information about the LTS.

Chapter 14

Testing in the cmi5 LMS Test Suite (LTS)

In this chapter:

- The importance of testing systems for cmi5 conformance
- About the cmi5 LMS Test Suite (LTS)
- Information about the user documentation and installation guidance

14.1 Purpose

This chapter is intended to provide an overview of the cmi5 LMS Test Suite (LTS), including the importance of testing systems for cmi5 conformance in a test suite, about the LTS, and where to find detailed installation and user guidance. The scope of this chapter focuses on utilizing the LTS to test systems for cmi5 conformance only and does not address testing requirements for cmi5 content (see Chapter 9 Testing in the cmi5 Content Test Suite), testing an LRS for xAPI conformance, or for testing systems that support other e-learning standards.

14.1.1 How to Use This Chapter

This section should be interpreted as an introduction to the LTS that was developed as part of the cmi5 CATAPULT project and explanation of why testing in the test suite is vitally important in terms of cmi5 conformance.

To match the terminology used in the cmi5 specification, LMS is used to describe the system that is "launching" the cmi5 content, but the system may or may not be a traditional Learning Management System.

14.1.2 Who This Chapter Is For

This chapter is for LMS professionals and systems developers looking to use the LTS to test and validate an LMS for cmi5 conformance. This chapter is also for program managers or LMS administrators who desire to test and validate the LMS before procurement.

14.2 The Importance of Testing an LMS for cmi5 Conformance

When building or procuring a cmi5 conformant LMS that will be integrated into a learning and training ecosystem, it is highly beneficial to test and validate that the LMS adheres to the cmi5 specification and will be compatible with cmi5 learning activities (content) and to other systems. The LTS allows systems developers and professionals to test and validate the LMS to ensure cmi5 conformance and plug and play interoperability with other systems in a learning ecosystem. Not using the LTS to test the developed LMS could result in interoperability issues. Without testing in the LTS, the developer won't know what is causing conformance issues (the learning activity or the LMS itself) should they arise. In addition, the LMS should be regularly tested, as the test pipeline allows, to ensure that the LMS continues to be conformant to the cmi5 specification.

The LTS is also a valuable tool for LMS professionals looking to procure a cmi5 conformant LMS. Having the vendor provide the JUnit XML output file from the LTS before procurement ensures that the LMS is conformant with the cmi5 specification. During the acquisitions process, the LMS vendor will also need to provide proof that they are following the best practice of regularly testing the LMS with each new release to ensure continued compliance with the cmi5 specification.

14.3 About the cmi5 LMS Test Suite (LTS)

The LTS application is a command line interface (CLI) that leverages the library of test packages to automate the testing of an LMS claiming to have cmi5 capability. The command line interface will expose an LMS integration interface that will have to be implemented on a per LMS basis.

The LTS is intended to be used by anyone needing to validate cmi5 conformance of an LMS or launching system, which may include developers, vendors, or those acquiring cmi5 enabled LMSs or launching systems.

To help developers or LMS professionals with the testing process, there is a procedure guide for navigating the testing process manually. Alternatively, a script can be written to automate the process. If one of the steps of the testing process does not produce the desired results as outlined in the procedure guide, then the developer will know that the LMS is in violation of that part of the cmi5 specification.

14.3.1 User Documentation Information

User documentation is available to help LMS developers and professionals navigate the LTS tool and contains links to the procedure document. The user documentation includes information on how to:

- Build the test package library
- Manually run the test suite
- Automatically run the test suite using a written script
- Output files

The LTS user documentation can be found here: https://adlnet.github.io/CATAPULT

Endnotes

Chapter 1: Modernizing Learning Ecosystems and the Role of cmi5

1: Advanced Distributed Learning Initiative, (n.d.). *cmi5 Specification.* https://adlnet.gov/projects/cmi5-specification/

2: Advanced Distributed Learning Initiative, (n.d.). *Total Learning Architecture.* https://adlnet.gov/guides/tla/

3: Advanced Distributed Learning Initiative, (n.d.). *Department of Defense Instruction 1322.26 ("Distributed Learning").* https://adlnet.gov/policy/dodi/

4: McDonald, B. (2015, August). *Time to plugin to cmi5?* [Article] LinkedIn. https://www.linkedin.com/pulse/time-plugin-cmi5-bill-mcdonald

5: Advanced Distributed Learning Initiative, (n.d.). *cmi5 Specification.* Retrieved May 21, 2021. https://adlnet.gov/projects/cmi5-specification/

Chapter 2: The Evolution of E-learning Standards

1: Advanced Distributed Learning Initiative, (n.d.). *SCORM® Overview.* https://adlnet.gov/projects/scorm/

2: Office of the Under Secretary of Defense for Personnel and Readiness. (2017). *DOD Instruction 1322.26 Distributed Learning (DL). Department of Defense.*

3: Advanced Distributed Learning Initiative, (n.d.). *cmi5 Specification.* https://adlnet.gov/projects/cmi5-specification/

4: Advanced Distributed Learning Initiative, (n.d.). *SCORM® Overview.* https://adlnet.gov/projects/scorm/

5: Scorm.com, (n.d.). *SCORM Explained 101: An introduction to SCORM.* https://scorm.com/scorm-explained/one-minute-scorm-overview/

6: United States Navy. (n.d.) *xAPI Contracting Requirements. xAPI Library.* https://netc.usalearning.net/xapi-library/acquisition-requirements-elearning.html

7: Johnson, A. & Hruska, N. [Advanced Distributed Learning Initiative]. (2013, Jan. 24). *The Experience API origins and capabilities webinar* [Video]. Streaming Service. https://www.youtube.com/watch?v=e8EDnoasTAg&t=361s

8: Advanced Distributed Learning Initiative & U.S. Department of Defense, (2013). *Experience API Specification.* https://github.com/adlnet/xAPI-Spec/blob/master/xAPI-About.md

9: Advanced Distributed Learning Initiative & U.S. Department of Defense, (2013). *Experience API Specification.* https://github.com/adlnet/xAPI-Spec/blob/master/xAPI-About.md

10: United States Navy. (n.d.) *xAPI Contracting Requirements. xAPI Library.* https://netc.usalearning.net/xapi-library/acquisition-requirements-elearning.html

11: Advanced Distributed Learning Initiative, (2020). *Defense Advanced Distributed Learning Advisory Committee 2020 Annual Report.*

12: Advanced Distributed Learning Initiative, (n.d.). *xAPI Profiles.* https://github.com/adlnet/xapi-profiles

13: xAPI.com, (n.d.). *xAPI Explained: Overview.* https://xapi.com/overview/

Chapter 3: cmi5: Transitioning to a Modern Learning Ecosystem

1: Office of the Under Secretary of Defense for Personnel and Readiness. (2017). *DOD Instruction 1322.26 Distributed Learning (DL).* Department of Defense.

2: Advanced Distributed Learning Initiative, (n.d.). *An Introduction to cmi5: Next-generation of e-Learning Interoperability.* https://adlnet.gov/resources/cmi5-resources/

3: Aviation Industry CBT Committee, (n.d.). *The cmi5 Project Overview.* https://aicc.github.io/CMI-5_Spec_Current/

4: Aviation Industry CBT Committee, (n.d.). *The cmi5 Project Overview.* https://aicc.github.io/CMI-5_Spec_Current/

5: Johnson, A., Miller, B., & McDonald, B. [Advanced Distributed Learning Initiative]. (2021, Mar.17.) cmi5 *Overview and The Way Ahead webinar.* [Video]. Streaming Service. https://youtu.be/JSIXnw3hoOg

6: Aviation Industry CBT Committee, (n.d.). *The cmi5 Project: Conceptual Overview of cmi5.* https://aicc.github.io/CMI-5_Spec_Current/flows/cmi5-overview.html

7: Aviation Industry CBT Committee, (n.d.). *cmi5 Specification Profile for xAPI.* [Section 5.0 Conceptual Model: Informative] https://github.com/AICC/CMI-5_Spec_Current/blob/quartz/cmi5_spec.md#50-conceptual-model-informative

8: Aviation Industry CBT Committee, (n.d.). *cmi5 Specification Profile for xAPI.* [Section 7.0 AU Requirements]. https://github.com/AICC/CMI-5_Spec_Current/blob/quartz/cmi5_spec.md#70-au-requirements

9: Aviation Industry CBT Committee, (n.d.). *cmi5 Specification Profile for xAPI* [Section 14.0 Course Package]. https://github.com/AICC/CMI-5_Spec_Current/blob/quartz/cmi5_spec.md#140-course-package

10: Welder IV, D. (2020). cmi5: *All About Blocks.* RISC, Inc. https://risc-inc.com/cmi5-all-about-blocks/

11: Johnson, A., Miller, B., & McDonald, B. [Advanced Distributed Learning Initiative]. (2021, Mar.17.) *cmi5 Overview and The Way Ahead webinar.* [Video]. Streaming Service. https://youtu.be/JSIXnw3hoOg

12: xAPI.com, (n.d.). *xAPI Tech Overview.* https://xapi.com/tech-overview/#pt3

13: xAPI.com, (n.d.). *cmi5: Technical 101.* https://xapi.com/cmi5-technical-101/

14: Aviation Industry CBT Committee, (n.d.). *The cmi5 Project Overview* [Section 9.3 Verbs]. https://github.com/AICC/CMI-5_Spec_Current/blob/quartz/cmi5_spec.md#verbs xAPI.com, (n.d.). cmi5 and the Experience API. https://xapi.com/cmi5/

15: xAPI.com, (n.d.). *cmi5 and the Experience API.* https://xapi.com/cmi5/

16: Advanced Distributed Learning Initiative, (n.d.). *An Introduction to cmi5: Next-generation of e-Learning Interoperability.* https://adlnet.gov/resources/cmi5-resources/

17: Aviation Industry CBT Committee, (n.d.). *The cmi5 Project Overview.* https://aicc.github.io/CMI-5_Spec_Current/

18: Werkenthin, A. (2020). *Why you should want cmi5.* RISC, Inc. https://risc-inc.com/cmi5-all-about-blocks/

19: Aviation Industry Computer-Based Training Committee, (n.d.). *SCORM vs cmi5 Comparison.* https://aicc.github.io/CMI-5_Spec_Current/SCORM/

19: Aviation Industry Computer-Based Training Committee, (n.d.). *xAPI/TinCan Package vs. cmi5 Comparison* http://aicc.github.io/CMI-5_Spec_Current/tincan

Chapter 4: cmi5 Implementation Guidance

1: Advanced Distributed Learning Initiative, (n.d.). *cmi5 Specification Matures to Support Online Learning Beyond SCORM®.* https://adlnet.gov/news/2021/07/20/cmi5-Specification-Matures/

2: Advanced Distributed Learning Initiative, (n.d.). *DoDI 1322.26 Fungible References.* https://adlnet.gov/policy/fungible/

3: Advanced Distributed Learning Initiative, (n.d.). *DoDI 1322.26 Fungible References.* https://adlnet.gov/policy/fungible/

Chapter 5: Deep Dive Into cmi5 Content

1: Aviation Industry CBT Committee, (n.d.). *cmi5 Specification Profile for xAPI* [Section 14.0 Course Package]. https://github.com/AICC/CMI-5_Spec_Current/blob/quartz/cmi5_spec.md#course_package

2: Welder IV, D. (2020). *cmi5: All About Blocks.* RISC, Inc. https://risc-inc.com/cmi5-all-about-blocks/

3: Aviation Industry CBT Committee, (n.d.). *cmi5 Specification Profile for xAPI* [Section 5.0 Conceptual Model: Informative]. https://github.com/AICC/CMI-5_Spec_Current/blob/quartz/cmi5_spec.md#50-conceptual-model-informative

4: Aviation Industry CBT Committee, (n.d.). *cmi5 Specification Profile for xAPI* [Section 13.1.2 Block Metadata]. https://github.com/AICC/CMI-5_Spec_Current/blob/quartz/cmi5_spec.md#1312-block-metadata

5: Aviation Industry CBT Committee, (n.d.). *cmi5 Specification Profile for xAPI* [Section 9.3 Verbs]. https://github.com/AICC/CMI-5_Spec_Current/blob/quartz/cmi5_spec.md#verbs

6: Aviation Industry CBT Committee, (n.d.). *cmi5 Specification Profile for xAPI* [Section 10.2.3 launchParameters]. https://github.com/AICC/CMI-5_Spec_Current/blob/quartz/cmi5_spec.md#1023-launchparameters

7: Aviation Industry CBT Committee, (n.d.). *cmi5 Specification Profile for xAPI* [Section 7.1.3 Types of Statements]. https://github.com/AICC/CMI-5_Spec_Current/blob/quartz/cmi5_spec.md#713-types-of-statements

8: Advanced Distributed Learning Initiative, (n.d.). *DoDI 1322.26 Fungible References.* https://adlnet.gov/policy/fungible/

Chapter 6: Acquiring cmi5 Content and Content Development Tools

1: Advanced Distributed Learning Initiative, (n.d.). *DoDI 1322.26 Fungible References.* https://adlnet.gov/policy/fungible/

2: Aviation Industry CBT Committee, (n.d.). *cmi5 Specification Profile for xAPI.* https://github.com/AICC/CMI-5_Spec_Current/blob/quartz/cmi5_spec.md

3: Aviation Industry CBT Committee, (n.d.). *The cmi5 Project Best Practices.* https://aicc.github.io/CMI-5_Spec_Current/best_practices/

Chapter 8: Building and Migrating cmi5 Content

1: Aviation Industry CBT Committee, (n.d.). *cmi5 Specification Profile for xAPI.* https://github.com/AICC/CMI-5_Spec_Current/blob/quartz/cmi5_spec.md

2: Aviation Industry CBT Committee, (n.d.). *The cmi5 Project Best Practices.* https://aicc.github.io/CMI-5_Spec_Current/best_practices/

3: Aviation Industry CBT Committee, (n.d.). *The cmi5 Project Common Mistakes.* https://aicc.github.io/CMI-5_Spec_Current/mistakes/

Chapter 10: Deep Dive Into cmi5 Systems Requirements

1: Aviation Industry CBT Committee, (n.d.). *cmi5 Specification Profile for xAPI* [Section 14.0 Course Package]. https://github.com/AICC/CMI-5_Spec_Current/blob/quartz/cmi5_spec.md#140-course-package

2: Aviation Industry CBT Committee, (n.d.). *cmi5 Specification Profile for xAPI* [Section 13.1.4 moveOn]. https://github.com/AICC/CMI-5_Spec_Current/blob/quartz/cmi5_spec.md#moveon

3: Aviation Industry CBT Committee, (n.d.). *cmi5 Specification Profile for xAPI*[Section 8.0 Content Launch Mechanisms]. https://github.com/AICC/CMI-5_Spec_Current/blob/quartz/cmi5_spec.md#80-content-launch-mechanisms

4: Aviation Industry CBT Committee, (n.d.). *cmi5 Specification Profile for xAPI* [Section 10.0 xAPI State Data Model]. https://github.com/AICC/CMI-5_Spec_Current/blob/quartz/cmi5_spec.md#100-xapi-state-data-model

5: Aviation Industry CBT Committee, (n.d.). *cmi5 Specification Profile for xAPI* [Section 8.1.2 fetch]. https://github.com/AICC/CMI-5_Spec_Current/blob/quartz/cmi5_spec.md#812-fetch

6: Aviation Industry CBT Committee, (n.d.). *cmi5 Specification Profile for xAPI* [Section 10.2.6 returnURL]. https://github.com/AICC/CMI-5_Spec_Current/blob/quartz/cmi5_spec.md#1026-returnurl

7: Aviation Industry CBT Committee, (n.d.). *cmi5 Specification Profile for xAPI* [Section 9.3.6 Abandoned]. https://github.com/AICC/CMI-5_Spec_Current/blob/quartz/cmi5_spec.md#936-abandoned

8: Aviation Industry CBT Committee, (n.d.). *cmi5 Specification Profile for xAPI* [Section 9.3.7 Waived]. https://github.com/AICC/CMI-5_Spec_Current/blob/quartz/cmi5_spec.md#937-waived

Chapter 11: Acquiring cmi5 Systems

1: Advanced Distributed Learning Initiative, (n.d.). *DoDI 1322.26 Fungible References.* https://adlnet.gov/policy/fungible/

2: Aviation Industry CBT Committee, (n.d.). *cmi5 Specification Profile for xAPI.* https://github.com/AICC/CMI-5_Spec_Current/blob/quartz/cmi5_spec.md

3: Aviation Industry CBT Committee, (n.d.). *The cmi5 Project Best Practices.* https://aicc.github.io/CMI-5_Spec_Current/best_practices/

Chapter 13: Building cmi5 Systems

1: Aviation Industry CBT Committee, (n.d.). *cmi5 Specification Profile for xAPI.* https://github.com/AICC/CMI-5_Spec_Current/blob/quartz/cmi5_spec.md

2: Aviation Industry CBT Committee, (n.d.). *The cmi5 Project Best Practices.* https://aicc.github.io/CMI-5_Spec_Current/best_practices/