Tool Evaluation Report

Learning Platform Enhancements Project

Project: Learning Platform Enhancements

Team: Joel Hillmann | Shubham Paudel | Mason Scanlan | Madison Tana

Client: Andy Connor, Customer Education Manager

Foster Moore, The Registry People®.

Mentor: Cheryll Singh

Date: [10 March 2025]

Version: 1.0

Contents

1.0 Introduction

The purpose of this report is to evaluate four SCORM-compliant authoring tools -LUMI, Xerte, eXeLearning and Adapt Authoring Tool -to determine the most suitable option for enhancing Foster Moore's Learning Management System (LMS). The evaluation criteria focused on SCORM compatibility, completion tracking, progress tracking, LMS interaction, browser compatibility, performance, user experience and customizability. This report outlines our findings and provides a recommendation based on our assessment.

2.0 Evaluation Criteria and Summary of Findings

Criteria	LUMI	Xerte	eXeLearning	Adapt Authoring
SCORM Version Compatibility: Does the tool support and properly export SCORM 1.2?	✅ 3 - Supports SCORM 1.2 but relies on iframes.	❌ N/A - Unable to verify due to login issues.	✅ 3 - Supports SCORM 1.2 but has issues with proper completion tracking.	❌ N/A - Not tested due to installation failure.
Completion Tracking: Does the LMS accurately track completion, especially for embedded SCORM content (not in pop-up windows)?	⚠️ 2 - Manual completion triggers needed.	❌ N/A - Not tested due to access issues.	⚠️ 2 - SCORM packages require manual fixes.	❌ N/A - Not tested due to installation failure.
Progress Tracking: Can learners' progress (including quiz attempts and interactions) be resumed within the LMS?	⚠️ 2 - H5P -based tracking limitations.	❌ N/A - Not tested due to access issues.	⚠️ 3- Progress tracking works, but needs debugging.	❌ N/A - Not tested due to installation failure.
Interaction with LMS: How effectively does the SCORM package communicate with TalentLMS to report progress, completion, and interaction data?	⚠️ 2 - Requires extra configuration	❌ N/A - Could not validate LMS interactions due to access issues.	⚠️ 3 – Some modification required for proper tracking.	❌ N/A - Could not validate due to installation failure.

Browser Compatibility: Does the SCORM package function consistently across different browsers without loss of interaction or tracking?	✅ 3 - Works in most modern browsers.	❌ N/A - Not tested due to access issues.	✅ 3 - Works in most modern browsers.	❌ N/A - Not tested due to installation failure.
Performance and User Experience: Is the SCORM package performing well within the LMS, and does it provide an intuitive user experience? Are there any usability issues like learners getting stuck?	⚠️ 2 - Completion issues can cause users to get stuck.	❌ N/A - Could not evaluate due to login issues.	⚠️ 2 - Users occasionally get stuck due to completion tracking issues, require manual fixes.	❌ N/A - Not tested due to installation failure.
Customizability: How easily can the tool be customized to include HTML, CSS, JavaScript, and interactive elements?	❌ 3 -Limited to H5P customizability.	❓ N/A - Documentation suggests customizability but could not be tested due to login issues.	✅ 4 – Supports custom HTML, CSS, JavaScript, and iDevices.	❓ N/A - Highly customizable, but installation issues prevented testing.
Overall Score Based on Rating: (Maximum Total 35)	✅ **17/35**	❌ **N/A**	✅ **20/35**	❌ **N/A**

3.0 Key Findings

3.1 LUMI

- **Strengths:** Easy to use, supports H5P-based interactions, widely compatible.
- **Weaknesses:** SCORM tracking is unreliable, iframe-based embedding causes security concerns, limited customization beyond H5P.
- **Suitability:** Better for simple, less customized interactive content.

3.2 Xerte

- **Strengths:** Open-source tool with potential customizability based on documentation.

- **Weaknesses:** Significant login and access issues prevented installation and evaluation.
- **Suitability:** Not viable for this project due to technical access challenges.

3.3 eXeLearning

- **Strengths:** SCORM 1.2 compatibility, progress tracking capable after modifications, with strong customization.
- **Weaknesses:** Completion tracking requires manual fixes, minor browser inconsistencies.
- **Suitability:** Best suited for this project due to its flexibility, ease of installation, and support for custom interactive elements.

3.4 Adapt Authoring Tool

- **Strengths:** Highly customizable based on research and documentation.
- **Weaknesses:** Complex installation process that failed despite multiple attempts.
- **Suitability:** Not viable for this project due to installation challenges.

4.0 Recommendation

Based on our evaluation, eXeLearning is the best choice for this project. While it does require some modifications to improve completion tracking, it provides:

- reliable SCORM 1.2 support.
- Good progress tracking after modification.
- Strong customizability, allowing us to develop complex interactive learning content.
- A straightforward installation process.

Although LUMI was considered, its iframe-based security limitations and restricted customization made it less suitable for our needs.

5.0 Next Steps

1. **Implement Fixes for eXeLearning's Completion Tracking -**Modify the SCORM completion handling in the SCOFunctions.js file.
2. **Develop Custom iDevices -**Leverage eXeLearning's extensive customizability to create interactive learning experiences.
3. **Conduct Final Testing -**Ensure courses created utilizing new iDevices are exported from eXeLearning as SCORM packages and function correctly (with completion tracking) within the TalentLMS.

6.0 Conclusion

This report provides justification for selecting eXeLearning as the authoring tool for SCORM content development. Despite some minor limitations, it offers the best balance of SCORM compatibility, customizability, and ease of integration. Moving forward, we will focus on optimizing eXeLearning's SCORM completion tracking and developing interactive learning modules using its customization capabilities.

7.0 Appendices

7.1 Additional Setup Instructions

To ensure the new iDevices and SCORM functions work properly:

1. **Install the Correct Version of eXeLearning:**
   - Download and install the full *install version* of eXeLearning (not portable or ready to run versions).
   - This ensures the application registers correctly with system paths and dependencies.
2. **Administrator Access:**
   - You must have **administrator rights** on the computer to edit or add files to protected directories like C:\Program Files (x86)\exe\scripts\idevices.
3. **After Updating or Adding iDevices:**
   - Clear the eXeLearning application cache stored in: APPDATA\exe
   - Restart eXeLearning. This ensures the program loads the updated or newly added iDevices correctly.

7.Source Code Modifications

File: SCOFunctions.js

Directory: In scripts directory within eXeLearning installation path (e.g., C:\Program Files (x86)\exe\scripts)

Instruction:

1. Replace the function unloadPage(isSCORM) with the following
2. add the finish course function
3. Ensure goBack() and goForward() functions are included.

```
// Mark the course as completed and successful
function finishCourse() {
  computeTime();

  if (typeof pipwerks !== "undefined" && pipwerks.SCORM) {
      // Mark this SCORM package as completed and passed
      pipwerks.SCORM.SetCompletionStatus("completed");
      pipwerks.SCORM.SetSuccessStatus("passed");

      // Save and quit SCORM session
      pipwerks.SCORM.save();
      pipwerks.SCORM.quit();
  } else {
      console.warn("SCORM API not available. Unable to set completion
status.");
  }
}

function unloadPage(isSCORM) {
  if (typeof isSCORM === "undefined") {
      isSCORM = false;
  }

  if (exitPageStatus !== true) {
      if (scorm.GetCompletionStatus() !== "completed") {
          scorm.SetCompletionStatus("incomplete"); // Ensure incomplete if not
finished
          scorm.SetSuccessStatus("failed");
      }
      doQuit();
  }
}

function goBack() {
  pipwerks.nav.goBack();
}

function goForward() {
  pipwerks.nav.goForward();
}
```

7.3 Original code from SCOFunctions

```
function unloadPage(isSCORM)
{

    if (parent && !parent.mod_scorm_is_window_closing){
        // #505 Issue
        parent.mod_scorm_is_window_closing = true
    }

    if (typeof isSCORM == "undefined"){
        isSCORM = false;
    }
    //console.trace('exitPageStatus'+exitPageStatus);

    var status;
    if (exitPageStatus != true)
    {
        status = scorm.GetSuccessStatus();
        // In SCORM12, information about completion and success is stored in
the same place (cmi.core.lesson_status)
        if (status!="passed" && status!="failed")
        {
            if(isSCORM==true)
            {
                scorm.SetCompletionStatus("incomplete");
                scorm.SetSuccessStatus("failed")
            }
            else
            {
                scorm.SetCompletionStatus("completed");
                scorm.SetSuccessStatus("passed")
            }
        }
        doQuit();
    }

    // NOTE:  don't return anything that resembles a javascript
    //        string from this function or IE will take the
    //        liberty of displaying a confirm message box.
}
```

7.4 New iDevices

How to Create an Idevice

1. **Duplicate an Example Idevice**:
    - To simplify the setup, copy an existing idevice folder (e.g., “example-idevice”) and rename it.
    - Ensure that the new folder structure includes config.xml, edition, and export.
2. **Edit config.xml**:
    - Open config.xml and update the idevice’s name, description, and other identifiers to reflect the new idevice’s purpose.
    - This customization allows eXeLearning to recognize the idevice as a unique option.
3. **Develop JavaScript/CSS in the edition and export Folders**:
    - Customize the JavaScript in the edition folder for editing functionality and in the export folder for published interactivity.
    - Include any CSS required for specific styling.

4. **Restart eXeLearning and clear AppData/exe**:
    - After creating and configuring the idevice, clear your cache in AppData/exe and restart eXeLearning to load the new idevice. It should now appear as an option in the idevice selection menu.

Folder Structure and Key Components of iDevices

1. **Config File (config.xml)**
    - This file is essential for defining the idevice's basic configuration. It contains metadata and settings such as the idevice's name, description, and icon.
    - The config.xml file also includes the structure that eXeLearning uses to recognize and display the idevice in its editor, allowing it to appear as a selectable option within eXeLearning's user interface.
2. **Edition Folder**
    - The edition folder holds the JavaScript (and CSS if needed) that controls the behavior and appearance of the idevice during the editing phase in eXeLearning.
    - The JavaScript file in this folder is responsible for generating the form fields or interactive elements that appear when an LMS author edits the idevice's content in eXeLearning.
    - **Example**: If the idevice includes a text input or a button, the JavaScript in the edition folder would define these elements and any custom functionality (e.g., character limits, validation, or dynamic responses).
3. **Export Folder**
    - The export folder contains the JavaScript and CSS files necessary for the idevice's functionality and styling when it is published to SCORM or exported from eXeLearning.
    - This folder ensures that the interactive features, such as buttons or progress tracking functions, work correctly on the LMS once the content is deployed.
    - **Example**: If the idevice includes a "Next" button that tracks completion, the JavaScript in the export folder would handle the completion status update and any additional interactions required for SCORM functionality.