UNITED STATES DEPARTMENT OF AGRICULTURE

USDA Content Integration Guide

Final

June 30, 2006

Prepared By

VERTEX Solutions, Inc.
7389 Lee Highway, Suite 301
Falls Church, VA 22042
POC: Dwayne Cotti (202) 694-0004
dwayne.cotti@usda.gov

Table of Contents

Tables

Appendix

Introduction

Purpose

The intent of this document is to provide developers of content for the United States Department of Agriculture (USDA) a better understanding of the USDA Information Technology (IT) environment, as well as the standards and requirements that apply to content developed for the USDA. This document is meant to be an overview and is not a substitute to any appropriate documentation regarding the specific technologies referenced herein. The technical specifications contained within this document are based on industry standards and best practices and are provided as guidance and references for designers and developers in the creation of content for the USDA.

Content developers should also note that the IT environment at USDA is constantly evolving, as are industry standard technologies and best practices. Contact the USDA's AgLearn Office with questions regarding the compatibility of content or practices with the USDA IT infrastructure.

Audience

It is assumed that content developers have knowledge of and/or experience with:

- Multiple delivery environments including UNIX web servers and CD-ROMs
- Industry standard web technologies including HTML, DHTML, XML, CSS, and JavaScript
- Information architecture and file management best practices including directory structures and the use of file naming conventions
- Techniques for creating accessible content meeting the requirements for Section 508 compliancy
- Requirements for creating Shareable Content Object Reference Model (SCORM) version 1.2 conformant content
- Learning Management System (LMS) integration

1.0 Environmental Considerations

It is important to consider how content will be delivered, managed, and potentially reused or repurposed. All content designed and developed for distribution within the USDA must allow for a variety of environmental variances without negatively impacting the user's viewing experience. These variances may include whether or not the content is to be delivered via the USDA Intranet, the Internet, or physical digital media such as CD-ROM or DVD-ROM. Some of the factors to consider when assessing the delivery environment include:

1. **Server Platform**

   All AgLearn servers used in staging and production are UNIX based servers and content should be developed with the UNIX platform in mind. UNIX is a case-sensitive platform, therefore case conventions and standards must be carefully followed, especially when naming and referencing files and directories.

2. **Network Bandwidth**

   When delivering courses across a network, identify the expected bandwidth and how it will affect any multimedia elements contained in the course. As a general best practice, it is recommended that all content be developed assuming delivery across a low bandwidth 56kbps connection. This will allow for the same content to be delivered both to users with higher bandwidth connections, and to those without network connections via a CD-ROM based delivery.

3. **Network Security**

   For purposes of security when dealing with SSI content, this access has been designed to incorporate a content caching capability and code has been implemented to clear cached content from the local computer cache at the end of each session.

   Presently, this solution works for SSI content in the HTML, SWF, PDF, and MS Office formats. For non-SSI courseware, caching is not a concern from a security standpoint; however the performance of this content is still affected by the caching implementation.

4. **Cross-Domain Scripting Issues**

   It is a known issue within the SCORM community that courses typically cannot be delivered by an LMS if the course content resides within another domain; a domain being defined as a web URL, not an IP address. For example, "www.abc.com" is not the same as "www.xyz.com". The source of this restriction is the JavaScript used to implement and support SCORM functionality, which is often not permitted for security reasons. It is necessary to fully understand this issue if the courseware is intended to be delivered or distributed across multiple domains. A complete description of this issue and possible solutions can be found in the document "Cross-Domain Scripting Issue, Version 1.0" from the ADL. This is not currently an issue for the USDA IT infrastructure as both the content and application reside within the same web domain, but it should be noted, as the platform is susceptible to change.

NOTE: For additional information on cross-domain scripting and SCORM, refer to the ADL website (http://www.adlnet.org). Additionally, cross-domain scripting is currently not allowed by USDA.

1.1 Host Platform Specifications

1. **Production Environment**

   Currently, all live and approved content reside within the USDA's production environment. No courseware testing can be administered on this server. A separate staging environment is configured to support content testing for both developers and for the USDA course owners. In addition, telecommunications impact assessments are also conducted within staging. No content will be migrated to the production environment until all phases of testing have been completed on staging.

2. **Staging Environment**

   Recently the AgLearn staging environment has been re-configured to support access from outside of the USDA firewall. This enables content developers to log into the system from external locations for the purposes of testing using generic login accounts. Team AgLearn should be contacted at teamAgLearn@usda.gov to obtain a generic login account. These accounts are shared and rotated between vendors as required. So do not consider your account assignment a permanent assignment. In addition, the production database has been migrated back to the staging environment. This allows USDA users and administrators access to the staging environment using their regular AgLearn e-Authentication logins.

1.2 End-User Platform Specifications

USDA End-Users will be using any combination of the following hardware, software, and OS to access AgLearn content.

1. **Operating Systems**

   a. Windows 95

   b. Windows 98

   c. Windows 2000 Professional with either SP2 or SP3 installed

   d. Windows XP Professional with SP1 installed

2. **Hardware**

   There are 29 sub-agencies within the USDA who maintain a variety of PC configurations. At a minimum, it is expected that

3. **Software**

   USDA computers may have a variety of software applications installed, the most common are:

   - Windows Media Player 8.0
   - WinZip 8.1
   - Adobe Acrobat Reader 5.0
   - Microsoft Reader 2.1
   - Shockwave Player 8.5
   - Flash Player 7

    - Microsoft Office
    - Java Runtime Environment (version varies but should be set to v1.4.2_07)

The minimum requirements for USDA PCs running content hosted in AgLearn is:

- Win32 Enabled
- Java Enabled
- Popup Blocking Disabled
- Javascript Enabled
- Cookies Enabled
- Screen Resolution 800x600 or greater
- Color depth 16 bit or greater
- Sun Java 1.4.2_07 or newer
- Flash 7.0 or newer

It is important to note and assume that most PC systems are “locked down” and end-users generally do not have access to certain administrative configuration rights or locations on the user’s hard drive. Known issues include but are not limited to:

- Users cannot write to the *C:\Program Files* directory
- Users cannot write to and/or may not have access to the *%systemroot%\Temp* directory
- Users cannot access the Downloaded Program Files directory
- Users cannot clear Internet Explorer cookies and/or temporary internet files
- Users cannot modify Internet Options including those associated with ActiveX objects or Java Virtual Machine settings

As a result of this “locked down” environment courseware should not include any custom programs that may need to be installed on a user’s computer as the user may lack the necessary administrative rights to complete this installation. Also, executable files or self-extracting files should be avoided, as the end user may not have the necessary administrative rights on the computer to successfully utilize their features or functionality. Any questions regarding user rights and/or permissions should be brought up prior to development so that they can be tested prior to content completion and delivery.

Another common setting that end-users generally do not have control over is the position and functionality of the Windows Taskbar. The taskbar, by default, is typically locked and set so that it does not auto-hide and it remains on top of other windows; therefore, content dimensions must take this factor into consideration. The maximum resolution recommended for courseware content is 775 pixels wide by 525 pixels high which should allow the entire content interface to be visible on screen under these conditions. It is recommended that contractors provide a sample of any new Graphical User Interfaces (GUIs) to be used prior to development so that it may be tested for proper display.

1.3 Plateau Learning Management System

The USDA AgLearn incorporates an implementation of the Plateau Learning Management System (LMS) Version 5.5. This LMS is based on a J2EE compliant architecture, supports the SCORM v1.2 standards, and is AICC certified. Tracking support in this LMS includes SCORM level 2, AICC level 2, and Plateau customized tracking features and functionality.

NOTE: For more information about the Plateau LMS visit the official Plateau website at http://www.plateau.com or view the Plateau CBT integration standards at http://content.plateausystems.com/contentintegration/index.htm

1.4 Configuration Management

Any changes outside of those involved in the normal enabling of content, including changes in architecture, operating system, hardware, software, players, etc, must be made by the USDA. The USDA must go through a review, testing, approval, and implementation process prior to any changes being made. The time involved in this process may be a prohibitive factor in the introduction of new technologies into the AgLearn.

2.0 SCORM Version 1.2 Conformance

The Shareable Content Object Reference Model (SCORM) has been developed as part of the Advanced Distributed Learning (ADL) Initiative and is designed to be a standard to promote the reusability and interoperability of learning content. Initially designed for use in the Department of Defense, SCORM has been adopted by the majority of LMS vendors as a supported standard, which has influenced its adoption among most other government agencies as well. The USDA AgLearn is no exception to this trend as the AgLearn's instance of the Plateau LMS supports SCORM version 1.2 conformant content. As a result, all content developed for inclusion in the USDA AgLearn shall be developed to be SCORM version 1.2 conformant.

While documentation on SCORM and its implementation is readily available, the guidance provided in the following sections is designed to be used as a brief reference of standards and best practices in the use of the SCORM in content developed for the AgLearn.

Note: For additional information on the implementation of the SCORM version 1.2 beyond that provided here, refer to the ADL website (http://www.adlnet.org).

2.1 Methods

The foundation of the SCORM is the communication between a Shareable Content Object (SCO) and the Learning Management System (LMS) or Learning Content Management System (LCMS), which hosts the content. This communication occurs through the use of eight ECMAScript (JavaScript) methods which are designed to initiate communication, set and retrieve information associated with any number of the data elements defined in the SCORM Run-Time Data Elements Model, terminate communication, and provide error-handling functionality in the event that this communication is unsuccessful at various levels.

Table 1: SCORM Methods

SCORM Methods
1. **LMSInitialize()** The LMSInitialize() method is designed to indicate to an LMS that the SCO wishes to initiate communication with it. This method can be used by the LMS to trigger any preparation necessary prior to the SCO passing and retrieving data element values. A SCO is allowed to make one and only one successful LMSInitialize() method call in a SCO session and the SCO must call this function before calling any other API functions with the exception of the three error handling methods (GetLastError(), GetErrorString(), and GetDiagnostic()). 2. **LMSFinish()** The LMSFinish() method is designed to indicate to the LMS when a SCO is finished communicating with it. At this time an LMS will also persist any data passed to it that has not yet been persisted. The LMS will not successfully invoke any calls subsequent to this function call with the exception of the three error handling methods (GetLastError(), GetErrorString(), and GetDiagnostic()).

SCORM Methods
3. LMSCommit() The LMSCommit() method triggers the LMS to persist any data that has been passed to the LMS by the SCO and has not yet been persisted. There are opportunities during which the LMSFinish() call may not occur as expected (such as power failure, system crashes, network errors, etc…) which could cause the permanent loss of data that is expected to be persisted. In an effort to decrease the likelihood of catastrophic failure due to one of these occurrences, it is recommended that the SCO implement an LMSCommit() method call at regular intervals or key instances so as to prevent any potential loss of learner data. **4. LMSGetValue()** The LMSGetValue() method provides the means for a SCO to retrieve the value assigned to a SCORM Data Element assuming that the element identified in the parameter passed to the LMSGetValue() method has been initialized with a value (either by the LMS or by the SCO) and is not identified as a write-only element. **5. LMSSetValue()** The LMSSetValue() method provides the means for a SCO to assign a value to a SCORM Data Element assuming that the element identified in the parameter passed to the LMSSetValue() method adheres to any rules for that particular data element regarding data type, range, and dependency; and the data element cannot be defined as read-only. **6. LMSGetLastError()** The LMSGetLastError() method is provided by SCORM as a means for a SCO to retrieve additional information with regards to why a method call may have failed. All method calls return an error message indicating the result of that call. The error will be “0” if a method executed successfully, or another characterstring representative of another specific result if the most recent method call was unsuccessful. **7. LMSGetErrorString()** The LMSGetErrorString() method provides additional basic information related to any of the particular error codes defined by SCORM or an individual LMS implementation. **8. LMSGetDiagnostic()** The LMSGetDiagnostic() method provides additional detailed explanation related to any of the particular error codes defined by SCORM or an individual LMS implementation. An LMS vendor primarily supplies this additional detail when an error can be attributed to a more specific cause.

2.2 Data Elements

The SCORM provides 14 mandatory data elements that must be supported by all SCORM version 1.2 conformant LMSs. While there are also a number of optional data elements that may be supported by an LMS, it is recommended that only the following mandatory elements be utilized to ensure interoperability of the developed content among future LMS instances that may or may not support any of the optional elements. The following table provides a brief reference to these mandatory elements and their intended use. Additional information can be found within ADL documentation.

Table 2: SCORM Mandatory Data Elements

SCORM mandatory Data Elements
1. cmi.core._children Read Only Contains a comma-delimited string of all of the elements in the core category that are supported by the LMS. An empty string is returned if an element is not supported or an element is supported and has no children
2. cmi.core.student_id Read Only Contains a unique case-insensitive alphanumeric identifier used by the LMS to identify the learner. The value may be up to 255 alpha-numeric characters in length with no spaces (hyphens and underscores are allowed, periods are not allowed)
3. cmi.core.student_name Read Only Contains the students official name as listed on the course roster, usually as a full name in the format: LastName, FirstName MiddleInitial
4. cmi.core.lesson_location Read / Write Intended to act as a location in which a SCO can store a "bookmark" in order to allow a student to return to a point in a SCO where they previously left off. The format is determined by the SCO and the LMS is indifferent to the format. The LMS does not interpret this value but simply makes it available to the SCO upon a user's return to that SCO.
5. cmi.core.credit Read Only Contains a value set by the LMS to indicate whether or not the student is viewing the content under the context of receiving credited based on his or her performance (pass/fail and/or score) in this SCO. The value shall be either "credit" or "no-credit" and is used in

SCORM mandatory Data Elements

conjunction with *cmi.core.lesson_mode* and/or *cmi.core.lesson_status*. A value of "no-credit" indicates to the SCO that any data sent by the SCO to the LMS will not change the student's accreditation

6. **cmi.core.lesson_status**

   Read / Write

   Intended to act as a location that stores the current student's status with regards to completion of the SCO. The value shall be "passed", "completed", "failed", "incomplete", "browsed", or "not attempted". This value may be set by the SCO or the LMS based on other data element values including *cmi.core.credit* and *cmi.core.lesson_mode*. **Note: AgLearn will not mark a SCO complete unless the "PASS" value is sent.**

7. **cmi.core.entry**

   Read Only

   Contains a value indicating whether or not a learner has previously viewed a SCO. The value shall be "ab-initio", "resume", or "" (an empty string). The value "ab-initio" indicates a user's first time viewing a SCO. The value "resume" indicates that a user is re-initiating a suspended SCO session. An empty string, "" indicates that a user has previously viewed a SCO but is re-entering under the context of not continuing a suspended session.

8. **cmi.core.score._children**

   Read Only

   Contains a comma-delimited string of all of the elements in the score category that are supported by the LMS. An empty string is returned if an element is supported and has no children, nothing is returned if an element is not supported

9. **cmi.core.score.raw**

   Read / Write

   Intended to act as a location to store a numerical value between 0 and 100 indicating the performance of a learner during his or her last attempt of the SCO.

10. **cmi.core.total_time**

    Read Only

    Contains a value indicating the total time a student has spent viewing a given SCO. This value is initialized by the LMS to equal 0 hours, 0 minutes, and 0 seconds with seconds optionally recorded to the hundredths (formatted as HHHH:MM:SS.SS) and the value is only increased by the session time *cmi.core.session_time* as it is passed to the LMS by the SCO upon the finish of a user session.

SCORM mandatory Data Elements

11. **cmi.core.exit**

    Write Only

    Intended to be used to pass a value to the LMS indicating the circumstances under which a learner has previously exited a SCO. The value shall be "time-out", "suspend", "logout", or "" (an empty string). This value should directly impact the value that the LMS will set *cmi.core.entry* to upon a user's next visit to the SCO and may affect the functionality of the LMS upon this subsequent visit.

12. **cmi.core.session_time**

    Write Only

    Intended to be used to pass a value to the LMS indicating the amount of time that the student has spent in this SCO session. This value includes hours, minutes, and seconds (with seconds optionally recorded to the hundredths) formatted as HHHH:MM:SS.SS.

13. **cmi.suspend_data**

    Read / Write

    Intended to act as a location to store any information that a SCO would like to persist until a subsequent session.

14. **cmi.launch_data**

    Read Only

    Contains any information that is to be provided to a SCO upon the start of any session of that SCO including a learner's initial visit. This value is initialized by the LMS to be equal to the information in the <adlcp:datafromlms> element in the manifest for the particular SCO.

2.3 Manifest

A SCORM manifest is an Extensible Markup Language (XML) description of a content object, a collection of content objects, a course (information about content objects as well as the content aggregation), or a collection of courses. For the purposes of the inclusion of content in the USDA AgLearn, a manifest should at a minimum contain the content aggregation of a course, any associated meta-data for that course, and the appropriate resource references identifying the SCOs and assets that make up the course.

The manifest file must be named *imsmanifest.xml* (case-sensitive) and be included with the content object files at the root of the directory containing these files. As part of the process of enabling content, a member of the AgLearn team will separate the manifest file from the rest of the content and compress it for uploading into the AgLearn. The AgLearn's implementation of Plateau limits the size of this file to 1MB once compressed.

NOTE: As dictated by the SCORM, only resources identified in the manifest as a SCO (those resources with their scormtype attribute set to "SCO" rather than "asset") will be able to communicate with the LMS. Developers should ensure that each resource's scormtype attribute is set to the appropriate value for the required functionality of that resource.

2.4 Meta-Data

The SCORM defines a number of meta-data elements for a variety of levels of a content package including the content aggregation itself, a SCO, an asset, and data types. USDA is not currently requiring the use of a specific set of meta-data elements, however it is expected that all mandatory meta-data elements be addressed.

NOTE: Although SCORM allows Meta-Data to be included either within a manifest itself or within a file referenced by the manifest; the practice adopted by the USDA AgLearn is the method of referencing an external Meta-Data file whenever Meta-Data is supplied. This helps in maintaining the portability of content as well as keeping the size of the manifest file to a minimum.

2.5 Tools for the Creation of SCORM Content

Third party tools are available to the developer or content creator to aid in the creation of SCORM conformant content as well as course manifests. In general, the market for tools of this nature is still fairly young. While the tools mentioned below can help in the creation of SCORM compliant content, none provides a complete, full feature-set solution.

1. **Macromedia Dreamweaver MX**

   Macromedia Dreamweaver is a web development tool that supports the inclusion of most web standards, including: HTML, JavaScript, ASP, XML, JSP, Flash, etc… This application is designed for general web-based content creation and website management. While Dreamweaver does not include any built in SCORM features, there are a number of extensions available and its features provide an excellent environment for efficient and consistent code development.

2. **Manifest Maker 2.0**

   Manifest Maker is an extension developed for Macromedia Dreamweaver that has the capability to scan a Dreamweaver site and generate XML manifest content, although only for one SCO at a time. It includes an interface for entering Meta-Data, scans directories and subdirectories for assets, and produces the applicable XML in a single manifest file.

3. **L5 SCORM Producer**

   L5 SCORM Producer is a Macromedia Dreamweaver extension for the organization and creation of manifest content. It has the capability to scan a Dreamweaver site and generate multiple manifest files for multiple SCOs. L5 SCORM Producer includes an interface for entering Meta-Data and creates a manifest for multiple SCO's; however, it does not automatically scan for assets.

4. **XMLSpy**

   XMLSpy is a robust tool for XML creation that, while not specifically SCORM focused, allows for the creation of custom meta-data entry interfaces, lending itself to being used within the realm of the SCORM.

5. **ADL Test Suite**

   The ADL Test Suite provides a means for developers to test for conformance with the SCORM when developing an LMS, a SCO, Meta-Data, and/or a Content Package. The test suite is available from the ADL website at http://www.adlnet.org.

6. **ADL Sample RTE**

   While the ADL Sample RTE (Run Time Environment) is not a full-featured LMS, it is an effective tool for the testing and validation of how a content package will function within an LMS environment. The ADL Sample RTE is available from the ADL website at http://www.adlnet.org.

3.0 Courseware Expectations

3.1 Performance

The usability of courseware content is of utmost importance, including clarity of the navigational structure, the means of navigating the content (the actual interface controls for navigation), accessibility both as mandated by Section 508 and as would be generally accepted in industry best practice, and any other aspects of the courseware that affects the user experience (such as utilization of appropriate SCORM features).

While the intent of the AgLearn is to deliver web-based training, lack of connectivity in some locations or the inaccessibility of a training room to a screener requires that all content be deliverable as both WBT and CBT. Some considerations of the dual purposing of content include limited scripting language support, file size optimization, and file type limitations.

Server-side scripting languages such as .php and .asp should be avoided in favor of JavaScript, which would allow the same content to be delivered on both a CD-ROM or over a web environment without any significant changes to the code. Content must not require Internet access to any remote files (such as those residing on a vendor's host server) since external Internet access may not be available in some locations. Content should load quickly over a range of bandwidths and network speeds including dial-up connections and LANs. Best practices for achieving this include the optimization of file types and sizes using the following techniques:

1. **Web-native content**

   For standard web-native content, which may include HTML, graphical elements, JavaScript scripting, etc, the total load not including any cached items (such as common graphical elements and shared JavaScript libraries) should not exceed 80KB in size. This would keep load times over a worst-case scenario of a 56k dial-up connection to approximately 10 seconds or less.

2. **GIFs, JPGs, and PNGs for graphical purposes**

   Any images used as part of course content should be in the .gif, .jpg, or .png format. Ideally the .png format is reserved only for use when the alpha transparency feature of a .png is absolutely necessary, as some older web browsers are not capable of displaying this file format. When using any of these file types for graphical purposes, it is recommended that their file size does not exceed 60KB in size and ideally remains smaller than 40KB in size.

3. **SWFs for content/animation/interactive graphics**

   A benefit to using Flash for courseware content includes the ability to break down an animation or interaction into multiple files that load individually to a user's computer. This speeds the initial appearance of the content and allows a developer to manipulate the efficiency of the animation's loading by dictating which externally loaded .swf files load and when. When using the .swf file format for graphical animation purposes, it is recommended that each individual file be a maximum of 70KB to 100KB in size.

4. **SWFs for audio/video**

   Multimedia content including audio and video should be optimized for delivery over low bandwidth connections. Currently, Macromedia Flash is the preferred format for multimedia type content being delivered over the USDA IT infrastructure. A benefit to using Flash is the ability to break down content into multiple files that load individually to a user's computer. It is recommended that no single .swf used for audio/video multimedia file be larger than 40MB in size.

5. **Other graphical/audio/video formats**

   Bandwidth intensive formats including .avi, .mpeg, or other audio/video formats that may significantly negatively impact the performance of the USDA network are generally not recommended for use in courseware content. If a new technique or format outside of those recommended by the USDA AgLearn is desired, a vendor should deliver content to act as a "preliminary technical test" for compatibility testing with the USDA infrastructure prior to the start of development work.

3.2 Features

The USDA requests that courseware content developed for delivery via the AgLearn utilizes the inherent functionality provided by the AgLearn's implementation of the Plateau LMS. Courseware is required to utilize the certificate functionality built into Plateau. The alternative for CD based course formats can include embedded written tests and/or printable certificates.

Other notable features desired of course content includes the tracking of content completion status, the tracking of student progress within a content object, and the recording of session time elapsed over a user's learning session. This functionality can be easily implemented in courses by taking advantage of the features provided by the SCORM.

All content designed for delivery via the AgLearn should incorporate the tracking of a user's status with regards to lesson completion. The SCORM provides the *cmi.core.lesson_status* data element for this purpose, therefore for consistency among courseware developed by various vendors, this is the desired means of reporting a user's completion of a content object to the LMS.

USDA also requests that all online courseware content incorporates some level of bookmarking capabilities. Often USDA employees are required to fulfill educational requirements over the course of multiple small lengths of time that may not be sufficient to complete an entire learning object. Bookmarking allows a user's progress to be stored so that the user may return to a point of learning at a later time. Courses should utilize the *cmi.core.lesson_location* SCORM Data Element to record bookmarks which are recommended to be set either at relevant points in the course content or at consistent intervals within the content.

1. **Bookmarking at relevant points in the course content:**

   Bookmarks can be placed at relevant points in the course content allowing a user to re-enter a content object at a sound restarting point, such as a topic introduction screen rather than in the middle of a topic. Over long lapses of time between viewing the content, returning a user to the middle of a topic may be confusing to a user or may not provide the necessary review of important content from screens earlier in a topic. For this reason, this bookmarking practice is favored over bookmarking at intervals.

2. **Bookmarking at consistent intervals throughout course content:**

   In the event that a SCO does not contain a secondary structure (such as topics) or obvious relevant points conducive to bookmarking, bookmarks can also be set at consistent intervals throughout the course content. This can be based on an approximate length of time or the number of screens passed since the last bookmark was set. By bookmarking at consistent intervals such as every 3 to 5 screens, a user's progress through potentially long content can be stored so that the user may continue where they left off when last visiting the content.

3. **Bookmarking at the user's point of exit:**

   Bookmarking only the screen that a user exits from is not a sufficient approach to achieve this persistence of progress as catastrophic failure (such as loss of communication with the LMS) may prevent the successful storage of this information.

Finally, for the purposes of reporting the effectiveness of the content being provided by the AgLearn, the SCORM *cmi.core.session_time* data element should be utilized whenever possible to record the elapsed time during which a user has spent reviewing the course material.

3.3 Testing and Evaluation

All courseware should be thoroughly tested and reviewed on all available platforms/client configurations. Content in any format other than WBT (including content to be delivered on CD-ROM, DVD-ROM, VHS, etc.) should also be delivered to the AgLearn team for testing of compatibility with the USDA IT infrastructure. Testing of WBT can be performed on any of the following environments.

Vendor Testing on the AgLearn Staging Environment

Content will be published by AgLearn for testing in the staging instance, which mimics the production environment. Content should be provided to the AgLearn team along with a content submission form for loading onto staging by USDA.

Vendor Testing on the Production Environment

No testing or vendor access to Production will be allowed. All critical course issues identified in AgLearn Production must be resolved through contact and coordinate with the USDA course owner and AgLearn.

3.4 Deliverables

WBT courseware must be delivered via a vendor hosted FTP site for purposes of quick turn-around, The content should be in a SCORM version 1.2 conformant zip package complete with a SCORM manifest. Contact teamaglearn@usda.gov for access to this FTP site.

Ensure that the content delivered for placement online includes only the required files for the display of the content. Files such as digital certificates, autorun.inf files, setup.exe files, installers, Photoshop and Flash graphics source files, or any other files not required for online delivery of the content must not be included in order to prevent the unnecessary migration of these files to the content servers. Files must not be Read-Only, but rather should allow for Full Control by the AgLearn team for successful uploading via Documentum.

Addendum I: Section 508 Compliance

In 1998, Congress amended Section 508 of the Rehabilitation Act to require Federal agencies to make their electronic and information technology accessible to people with disabilities. In general, all electronic media developed by or for the US government is required to adhere to Section 508 guidelines. However, there are certain areas where an exemption from Section 508 compliance is allowed. According to the memo titled “Guidance for Implementation of Congressionally Mandated Requirements Concerning Access to Federal Information Technology by Members of the Public and Federal Employees with Disabilities (Section 508 of the Rehabilitation Act)”, dated March 6, 2001:

> “Exemption from Section 508 compliance exists for several areas. This includes EIT used for telecommunications or information systems, the function, operation, or use of which involves intelligence activities, cryptologic activities related to national security, command and control of military forces, equipment that is an integral part of a weapon or weapons system, and systems which are critical to the direct fulfillment of military or intelligence missions. However, it is possible that some classified websites may still require compliance for the benefit of employees with disabilities. Also exempt is the installed base of EIT prior to 21 June 2001, i.e., Section 508 is not retroactive, only forward looking.”

Due to the nature of some of the SSI content being developed for the USDA AgLearn, it is possible that some content may be exempt from Section 508 compliance requirements, however this determination is not to be taken lightly or made arbitrarily. Consult with your USDA Project Manager to determine whether or not content being created must adhere to Section 508 guidelines.

NOTE: For more information on Section 508 beyond that provided here, visit the US Government’s Section 508 website at http://www.section508.gov/ or view the standard itself at http://www.access-board.gov/508.htm. The complete memo “Guidance for Implementation of Congressionally Mandated Requirements Concerning Access to Federal Information Technology by Members of the Public and Federal Employees with Disabilities (Section 508 of the Rehabilitation Act)” can be found at http://www.forscom.army.mil/6Mar01_DA.pdf.

Section 508 Provisions

Table 3 (below) address each of the provisions under the “Web-based Intranet and Internet Information and Applications (1194.22)” section of the “Section 508 Standards for Electronic and Information Technology.” A brief interpretation of each item is provided followed by suggested best practices for implementing the provision. Practical recommendations are also included for providing alternative access to web-based and multimedia content as applicable.

Table 3: Section 508 Provisions

SECTION 508 PROVISIONS
1. §1194.22(a) A text equivalent for every non-text element shall be provided (e.g., via “alt", "longdesc”, or in the element content). a. *Interpretation:* Images, animations, audio, and video are all examples of non-text elements. These items cannot be interpreted by assistive technology, so their meaning must also be available through an accessible text description.

SECTION 508 PROVISIONS

b. *Practices:*

- Include alt and title attributes for images including those used as list bullets, buttons, spacers, etc.
- If an image is used as a link include a title attribute in the link tag
- Use alt and title attributes in a dummy image to describe applets, objects, audio, or video.
- D links should be used to link to an accessible description if the required description is substantial in length

2. **§1194.22(b) Equivalent alternatives for any multimedia presentation shall be synchronized with the presentation.**

a. *Interpretation:*

A multimedia presentation appeals to both the visual and auditory senses. Text equivalents for this type of presentation must be synchronized so that the meaning of the content is not misrepresented.

b. *Practices:*

- Synchronize text equivalents of multimedia presentations.

3. **§1194.22(c) Web pages shall be designed so that all information conveyed with color is also available without color, for example from context or markup.**

a. *Interpretation:*

The use of black and red text to indicate positive and negative numbers is one example of using color to convey information. A colorblind individual might be unable to distinguish between the two text colors, so text or markup should be used to clarify this information.

b. *Practices:*

- Avoid the use of color to convey information whenever possible
- If color is necessary, clarify via an alt tag for images or markup (italics, underline, etc.) for text

4. **§1194.22(d) Documents shall be organized so they are readable without requiring an associated style sheet.**

a. *Interpretation:*

Style sheets can be used to control formatting of a web page; however, some individuals may wish to override these style sheets in order to modify font type, size, or color, or to change background colors for easier reading. This can cause objects on screen to become misaligned or text on screen to blend into the background among other potential problems. Therefore, documents must be designed in such a way that the same meaning is conveyed and the content is readable with or without the style sheet.

b. *Practices:*

- Verify that a webpage is viewable with or without the associated style sheet

SECTION 508 PROVISIONS

- Look for layered text disappearing over a background or other text
- Look for misalignment of screen elements such as the caption for one image being moved near another image

5. **§1194.22(e) Redundant text links shall be provided for each active region of a server-side image map.**

   a. *Interpretation:*

   A server-side image map may not reveal the destination of a link to a user, so redundant plain text links should be provided in addition to the image map.

   b. *Practices:*

   - Avoid the use of server-side image maps whenever possible
   - If server-side image maps are necessary, provide redundant plain text links

6. **§1194.22(f) Client-side image maps shall be provided instead of server-side image maps except where the regions cannot be defined with an available geometric shape.**

   a. *Interpretation:*

   A server-side image map may not reveal the destination of a link to a user, so a client-side image map should be used in place of a server-side image map whenever possible.

   b. *Practices:*

   - Avoid the use of server-side image maps using client-side image maps instead
   - Assign a title attribute to each image map link to indicate the purpose of that link

7. **§1194.22(g) Row and column headers shall be identified for data tables.**

   a. *Interpretation:*

   Some assistive technology relies on markup to indicate what a heading is and what data in a data table is. To prevent a user from becoming lost in a table, row and column headings should be indicated by markup.

   b. *Practices*

   - Use the scope attributes to identify a row or column heading
   - Use the <th> tag instead of the <td> tag for column headings
   - Use the <thead> tag instead of the <tr> around the column headings row

   c. *Sample:*

```
<table>
    <thead>
        <th scope="col" >Name</th>
        <th scope="col" >Phone Number</th>
    </thead>
    <tr>
```

SECTION 508 PROVISIONS

```
        <td scope="row" >John Doe</td>
        <td>(123) 456-7890</td>
    </tr>
</table>
```

8. **§1194.22(h) Markup shall be used to associate data cells and header cells for data tables that have two or more logical levels of row or column headers.**

    a. *Interpretation:*

    Tables containing more than one level of information should utilize markup to specify what category and subcategory the data belongs to.

    b. *Practices:*

    - Use the id attribute to identify a row or column heading
    - Use the headers attribute to specify which row and column the data belongs to

    c. *Sample:*

    ```
    <th id="name">Name</th>
    <th id="phone">Phone</th>
    <th id="address">Address</th>
    <td id="john">John Doe</td>
    <td headers="john phone">(123) 456-7890</td>
    <td headers="john address">123 Main St.</td>
    ```

9. **§1194.22(i) Frames shall be titled with text that facilitates frame identification and navigation.**

    a. *Interpretation:*

    Frames allow different components of a page to be located in separate files, such as content in one and navigation controls in another. A user might become lost in a website while they search one frame for navigation controls that actually appear in another.

    b. *Practices:*

    - Use the title attribute in each frame and assign a meaningful title

10. **§1194.22(j) Pages shall be designed to avoid causing the screen to flicker with a frequency greater than 2 Hz and lower than 55 Hz.**

    a. *Interpretation:*

    Blinking or flashing elements on a web page can be a danger to some individuals with certain epileptic conditions, especially those that flash at a frequency between 2 and 55 Hz. Effects or animations that might cause a screen to flicker between 2 and 55 Hz should be avoided.

    b. *Practices:*

SECTION 508 PROVISIONS

- Avoid animations that may cause screen flicker
- Measure animations that cause screen flicker to insure it is not between 2 and 55 Hz

11. **§1194.22(k) A text-only page, with equivalent information or functionality, shall be provided to make a web site comply with the provisions of this part, when compliance cannot be accomplished in any other way. The content of the text-only page shall be updated whenever the primary page changes.**

    a. *Interpretation:*

    In the event that a web page or element within a web page cannot be made to comply with the above conditions, an alternative compliant page should be created. This page should be kept up to date and always reflect the content of non-complaint page.

    b. *Practices:*

    - Use accessible alternative pages as necessary

12. **§1194.22(l) When web pages utilize scripting languages to display content, or to create interface elements, the information provided by the script shall be identified with functional text that can be read by assistive technology.**

    a. *Interpretation:*

    If a scripting language is used to dynamically change the content of a screen, the changed content should be recognizable by assistive technology.

    b. *Practices:*

    - Force focus on any dynamic object on screen to focus the assistive technology on it
    - Avoid SetTimeout( ) function calls whenever necessary as this can interfere with the assistive technology's ability to interpret the web page

13. **§1194.22(m) When a web page requires that an applet, plug-in or other application be present on the client system to interpret page content, the page must provide a link to a plug-in or applet that complies with 1194.21 (a) through (l).**

    a. *Interpretation:*

    If the content of a web page requires an applet or plug-in, a link to this applet or plug-in must be provided and that link must comply with the above sections regarding accessibility.

    b. *Practices:*

    - Provide a course requirements page with accessible links to any required plugins

14. **§1194.22(n) When electronic forms are designed to be completed on line, the form shall allow people using assistive technology to access the information, field elements and functionality required for completion and submission of the form, including all directions and cues.**

    a. *Interpretation:*

    If a web page contains a form or form objects, these objects must be accessible by a

SECTION 508 PROVISIONS

user of assistive technology. Items should be in a logical order with a rational tab order and all items should be labeled with text or markup to easily associate an item with its purpose.

b. *Practices:*

- Assign <label> elements to form elements
- Ensure items appear in logical order
- Set tab order of form items as needed but only when necessary

15. **§1194.22(o) A method shall be provided that permits users to skip repetitive navigation links.**

a. *Interpretation:*

Any page with repetitive links (such as navigation bars or links in a header or footer) should also contain a method to skip past these links. This would prevent a user utilizing a screen reader from having to listen to repeated links on every page of a site.

b. *Practices:*

- Include a hidden link prior to any repetitive links that shifts focus past the repetitive links

16. **§1194.22(p) When a timed response is required, the user shall be alerted and given sufficient time to indicate more time is required.**

a. *Interpretation:*

Any action that occurs after a specified amount of time with or without interaction from a user is considered a timed response including page refreshing or redirecting. In these instances, notification should be given to the user and the option made available to request more time to complete the action. This would prevent a user from missing any relevant information on a page due to having insufficient time to review the page content.

b. *Practices:*

- Avoid the use of timed responses whenever possible
- If a timed response is necessary, allow a user to request more time if needed

Evaluating Courseware for Accessibility

The following sections outline a process for evaluating content for conformance to Section 508 accessibility provisions.

1. **Evaluation During Development**

   Developers should use the best practices recommendations outlined in Section 2.1 Section 508 Provisions to ensure that programmed content is accessible. Periodically during development, developers should test various pages containing different types of content using tools such as Bobby, LIFT, and screen reading software. Identifying accessibility issues early reduces and avoids time spent later to correct content that cannot be accessed by users with disabilities or alternate browser software.

2. **Evaluation After Development**

   Once content development is complete, a preliminary conformance evaluation should be conducted using semi-automatic and manual checking methods. The following five-step process is excerpted from the World Wide Web Consortium Web Accessibility Initiative document, "Evaluating Web Sites for Accessibility."[1]

   a. Select a representative sampling of different kinds of pages from the Web content to be reviewed; must include all pages on which people are more likely to enter your content.

   b. Use a graphical user interface (GUI) browser (such as Internet Explorer, Netscape Navigator, or Opera) and examine the selection of pages to be assessed. Some of these manual checks can be performed by changing settings or preferences in the browser, some may require changes to operating system settings, and some may require additional software.

      - Turn off images, and check whether appropriate alternative text is available.
      - Turn off the sound, and make sure audio content is still available through text equivalents.
      - Use browser controls to vary font-size: verify that the font size changes on the screen accordingly; and that the page is still usable at larger font sizes.
      - Test with different screen resolution, and/or by resizing the application window to less than maximum, to verify that horizontal scrolling is not required (caution: test with different browsers, or examine code for absolute sizing, to ensure that it is a content problem not a browser problem)
      - Change the display color to gray scale (or print out page in gray scale or black and white) and observe whether the color contrast is adequate.
      - Without using the mouse tab through the links and form controls on a page, making sure that you can access all links and form controls, and that the links clearly indicate what they lead to.

   c. Use a voice browser (such as JAWS) or a text browser (such as Lynx) and examine the Web site while answering these questions:

[1] Copyright © 2001-2002 W3C (MIT, INRIA, Keio), All Rights Reserved. http://www.w3.org/Consortium/Legal/2002/copyright-documents-20021231

- Is equivalent information available through the voice or text browser as is available through the GUI browser?
- Is the information presented in a meaningful order if read serially?

d. Use two general accessibility evaluation tools and note any problems indicated by the tools.

e. Summarize results

- Summarize the types of problems encountered
- Indicate the method by which problems were identified
- Recommend follow-up steps and ways to address any identified problems.

3. **Final Verification**

   Once all identified accessibility problems have been addressed and corrected, the web content is ready to be tested and verified before being published to a live production environment such as the Plateau LMS.

Tools for Assessment and Correction of Accessibility

The following section of this document provides information about some of the available tools and processes more commonly used among web developers for evaluating and modifying web-based content to meet accessibility requirements.

1. **Macromedia Dreamweaver MX**

   Dreamweaver MX includes an accessibility reports feature to check that you have implemented the guidelines in your site. You can run an accessibility report on a current document, selected files, a folder, or an entire site.

2. **Bobby**

   Bobby is a comprehensive web accessibility software tool designed to help expose and repair barriers to accessibility and encourage compliance with existing accessibility guidelines. Bobby tests for compliance with government standards, including the U.S. Government's Section 508. It offers prioritized suggestions based on the Web Content Accessibility Guidelines provided by the World Wide Web Consortium's (W3C) Web Access Initiative. Bobby allows developers to test web pages and generate summary reports highlighting critical accessibility issues before posting content to live servers.

3. **Lift for Dreamweaver**

   Lift for Dreamweaver is an extension for the Macromedia Dreamweaver authoring tool. Lift allows customization of accessibility tests (WCAG Priority 1, US government section 508), and provides continuously updated test results during editing as well as a wizard-based repair tool that can be run in a context-sensitive mode while editing. It generates reports in HTML format or as an XML file that can be used for further processing. It incorporates heuristics for distinguishing different types of tables and images to provide appropriate suggestions for repair. LIFT is now fully integrated within Dreamweaver MX.

4. **AccMonitor**

   With AccMonitor, you can monitor the status of an entire Web site or a subdirectory of the site. Using the AccMonitor console, you can include or exclude directories or file types. You can send "email alerts" to any number of recipients. And when you monitor Web files, you can use Accessibility (Section 508 standards or W3C guidelines) report modes for the accessibility rules you choose, Searchability Rules, Privacy Rules, or any Test Suite that you develop or is shipped with AccMonitor.

5. **Site Valet**

   Site Valet is a comprehensive Quality Assurance product for Web and Intranet sites. An online toolkit is complemented by a site maintenance program, both of which are available to the public at the website. Site Valet can be extensively customized for corporate users. Accessibility Valet is a core Site Valet tool, designed to help ensure accessibility by analyzing markup for conformance to web accessibility guidelines: specifically the WCAG and Section 508.

6. **JAWS® for Windows**

   JAWS® for Windows provides access to today's software applications and the Internet. With its internal software speech synthesizer and the computer's sound card, information from the screen is read aloud, providing technology to access a wide variety of information, education and job related applications. JAWS also outputs to refreshable Braille displays, providing unmatched Braille support of any screen reader on the market. A training tutorial is included.

7. **Window-Eyes**

   Window-Eyes is a leading software application for the blind and visually impaired, which converts components of the Windows operating system into to synthesized speech allowing for complete and total access to Windows based computer systems. Window-Eyes integration into Windows is seamless, providing you with instant access to the operating system without having to learn a complicated set of keystrokes.

Appendix A List of Acronyms

ADL – Advanced Distributed Learning

API – Application Program Interface

CBT – Computer Based Training

COTS – Commercial Off the Shelf

GUI – Graphical User Interface

IT – Information Technology

LCMS – Learning Content Management System

LMS – Learning Management System

SCO – Shareable Content Object

USDA – United States Department of Agriculture

RTE – Run Time Environment

SCORM – Shareable Content Object Reference Model

WBT – Web Based Training

Appendix B Production Host Server Specifications

1. Application Server
   - TBD
2. Web Server
   - TBD
3. Database Server
   - TBD

Appendix C Staging Host Server Specifications

1. Application Server
   - TBD
2. Web Server
   - TBD
3. Database Server
   - TBD