



ADVANCED
DISTRIBUTED
LEARNING

CHOOSING AUTHORING TOOLS

March 10, 2009
VERSION 1.4



<http://creativecommons.org/licenses/by-nc-sa/3.0/>

This page intentionally left blank

Table of Contents

1. Purpose and scope of this paper	5
2. Overview.....	5
2.1 What is an authoring tool?	5
2.2 Why use authoring tools?.....	6
2.3 Why is the choice of tools so important?.....	6
2.4 Should my organization mandate use of standard tools?	6
3. Process for choosing tools.....	7
4. Categories and examples of authoring tools	8
4.1 Self-contained authoring environments.....	9
4.1.1 Web development tools.....	9
4.1.2 Rapid Application Development (RAD) tools	9
4.1.3 E-learning development tools.....	9
4.1.4 Simulation development tools	11
4.1.5 Game development environments	11
4.1.6 Virtual world development environments	11
4.1.7 Database-delivered web application systems	12
4.2 Learning content management systems (LCMSs)	12
4.3 Virtual classroom systems	12
4.4 External document converter/optimizer tools	13
4.4.1 Web-based external document converter/optimizer tools.....	13
4.4.2 Desktop based external document converter/optimizer tools	13
4.5 Auxiliary tools	14
4.5.1 E-learning assemblers/packagers	14
4.5.2 Specific interaction object creation tools	14
4.5.3 Media asset production tools.....	14
4.5.4 Word processors, page layout, and document format tools.....	15
4.5.5 Database applications	15
4.5.6 Web-based collaboration and web 2.0 authoring tools	15
4.5.7 Web page editors	15
5. Templates and skins.....	16
6. File formats	16
6.1 Input	16
6.2 Output	16
7. Standards support.....	17

7.1 SCORM.....	17
7.2 Section 508	19
8. Criteria for assessing quality and suitability of tools.....	20
9. General recommendations.....	22
10. Current Trends in Authoring Tools	23
10.1 Team-based life cycle production and maintenance.....	23
10.2 XML Output.....	23
10.3 Separation of content and appearance	23
10.4 Centralized control with distributed contribution.....	23
10.5 Templates and skins	24
10.6 Learning object-centric architecture	24
10.7 Embedded best practice design principles.....	24
10.8 Support for advanced learning technologies.....	24
11. For Further Reference	24
Appendix.....	25
Sample Tool Requirements Matrix.....	25
Sample Tool Features Rating Matrix	26

NOTE: Vendor citations or descriptions in this paper are for illustrative purposes and do not constitute an endorsement by ADL. All listings of vendors and products are in alphabetical order unless otherwise noted.

1. Purpose and scope of this paper

The purpose of this paper is to help those involved in the process of choosing authoring tools to make an informed decision. The paper presents a range of considerations for choosing tools, whether as an enterprise-wide acquisition or a single user purchase, and includes a sampling of current tools categorized according to the kind of product they are intended to produce.

This paper does not contain a comprehensive survey of available tools on the market, nor does it contain a comparative rating or evaluation of products, and should not be construed as such. For more in-depth information about tools and their features, see the references in section 11: *For your reference* or consult the vendors. ADL presents this paper merely as a guide to the issues, opportunities, and processes that are typically considered in choosing authoring tools.

ADL has titled this paper “Choosing Authoring Tools” rather than “Choosing an Authoring Tool,” since there is usually a need to select more than one product. Rarely will one tool meet all the production needs of an organization or developer. Most developers use a combination of tools, even to produce a single e-learning product; using a combination of tools that are each optimized to produce particular components of the product can increase production efficiencies dramatically. And you may find it impossible to create the variety of e-learning product types your organization requires with a single tool.

In line with our mission to promote reusability and interoperability in e-learning, ADL recommends authoring tools with built-in features that allow creating SCORM®-compliant e-learning. Creating e-learning that is not reusable or interoperable can be a significant business risk. You can find SCORM considerations for authoring tools in section 7.1: *SCORM*.

2. Overview

2.1 What is an authoring tool?

Authoring tools are software applications used to develop e-learning products. They generally include the capabilities to create, edit, review, test, and configure e-learning. These tools support learning, education, and training by enabling using distributed e-learning that is cost-efficient to produce, and that facilitates incorporating effective learning strategies and delivery technologies into the e-learning.

Authoring tools range from advanced software to create a wide array of sophisticated applications (not limited to e-learning), to simple tools that convert instructional PowerPoint slides to web pages. In this regard, it is important to understand that some software tools used as authoring tools are not necessarily designed for the creation of e-learning specifically; they can be open-ended, multi-purpose tools designed to create, for instance, any kind of web page/site. But when developers use them to create e-learning, they are referred to as authoring tools.

Vendors build some authoring tools into systems that perform broader functions; this is the case with learning content management systems (LCMSs). In many LCMSs, you can decouple the authoring tool component and use it as a separate application to develop and output e-learning without relying on the other components of the system.

As described in section 1: *Purpose and scope of this paper*, developers rarely use authoring tools in isolation; in fact, most developers use more than one software tool during the production process, and a

substantial number use four or more. Even when using a combination of tools, however, a developer generally uses one primary tool to create the base screens and assemble them into an e-learning product. These tools are distinguished from auxiliary software tools (for instance, Adobe Photoshop) that are not authoring tools but may be used in support of or in conjunction with those tools. This paper includes a discussion of auxiliary tools.

Authoring tools discussed in this paper refer to web-based e-learning (WBT); CD-ROM or DVD-based e-learning has largely disappeared due to standardizing intranets as browser-based, and distributing efficiencies using web-based delivery. However, many authoring tools offer disc-based delivery as an output option, to support environments where bandwidth is limited or non-existent.

2.2 Why use authoring tools?

Authoring tools (as opposed to writing code or script directly in a programming editor) reduce technical overhead; they generally use WYSIWYG (“what you see is what you get”) interfaces allowing users to easily manipulate and configure e-learning assets, using familiar visual metaphors. Thus, programming editors that facilitate writing application code like C++ or script languages like JavaScript are not truly authoring tools; developers can indeed use them to author e-learning, but they are not designed to reduce the technical overhead of knowing the programming or scripting language. Furthermore, most training organizations do not have the advanced (and expensive) programming skill sets in their development staff to program e-learning applications using only programming languages or scripts, and they do not have the infrastructure to support traditional software application development.

Primarily, authoring tools serve to reduce the skill set requirements for the authoring process, in some cases to a level where an untrained user can start using a tool and producing screens within minutes.

Secondarily, most authoring tools base a major part of their value-add proposition on automating time-consuming tasks, optimizing workflows, and generally offering a more streamlined and efficient approach to the authoring process, which can be very time consuming.

2.3 Why is the choice of tools so important?

Choosing e-learning authoring tools is one of the most crucial decisions any training organization, project, or developer can make. Authoring tools are designed for particular styles of learning, delivery platforms, file formats, e-learning standards, and production workflows. If your organization chooses a tool or set of tools that is not optimized for your needs, you could waste a lot of time and money creating e-learning that does not function correctly within your training infrastructure or is instructionally ineffective.

Another critical factor in choosing tools—one that can make or break an organization’s training budget with costly conversions—is durability. This relates to whether the tools will have longevity in the marketplace such that they continue to be available and supported, allowing source files to be opened and edited in future versions of the application. It also relates to whether the tools will, in the future, produce output formats supported by browser versions and browser plug-in updates.

2.4 Should my organization mandate use of standard tools?

Many organizations wonder whether they should mandate adopting a particular set of tools as a required standard across their organization. This has many advantages, among them:

- Reducing costs through purchase of group or enterprise licenses that lower the per end-user cost
- Providing for economies of scale in training to use the tools, help desk support, configuration management, etc.

- Making enforcing uniform standards easier through dissemination of application source file templates

The most important consideration in whether to standardize on tools, however, is the variability in types of training your organization produces. As stated above, authoring tools are optimized for particular types of training or IT environments. The worst thing you can do is mandate use of a single tool as the organizational standard. This can effectively amount to enforcing one style or type of training across the organization, which may be counter to the organization's (or even single project's) needs. More and more nowadays, training programs incorporate disparate elements in a blended or hybrid solution.

For instance, you may decide that the best way to teach some skills in a course is through asynchronous e-learning, while you may decide to teach other skills in the same training course through a synchronous virtual classroom environment. The choice of authoring tools probably will not be the same for both. You must take this into account in developing the tool standard specifications, when tools are standardized throughout the organization. The standard must address each type of learning, file output type, etc. with a standard tool specification for each type. Seldom will one tool suffice to cover all aspects of the authoring process or meet all needs for the various types of training produced by the organization.

Before specifying tool standards, ADL recommends that you standardize the requirements for the e-learning products themselves, using style guides and other policy documents. This includes such things as look and feel, interface functionality, file formats, course elements, and assessment design. This will drive and clarify the choice of tools.

In DoD, the MIL HDBK 29612 provides guidelines that may be useful in developing standards. Part 2 deals with training design and development, and Part 5 deals specifically with e-learning. Note that these are not mandates; they are guidelines only.

3. Process for choosing tools

ADL recommends the following high-level process for choosing authoring tools (you probably will need more than one tool to produce multiple kinds of training, interactions, etc.):

1. Determine requirements and parameters for the following:
 - Type(s) of training (sometimes multiple types are required in your organization)
 - Asynchronous e-learning
 - Synchronous virtual classroom or virtual world
 - Asynchronous virtual classroom (for example, recorded synchronous classroom sessions)
 - Instructor-led training (ILT) with certain aspects delivered electronically (for example, assessments)
 - Particular learning functions needed, especially social learning functions such as wikis, blogs, forums, and chat.
 - Media
 - Audio
 - Video
 - Graphics
 - 2D animation
 - 3D animation
 - Level of interactivity
 1. Passive – no interactivity except to navigate to next screen

2. Simple interactions limited to elaboration of information or getting feedback
 3. Adaptive navigation and branching
 4. Highly interactive simulation with granular assessment and adaptive learning paths
- Skill sets of authors
 - Need for non-technical staff to edit content (this is especially important where content changes frequently or client wants to take over course maintenance responsibilities)
 - Output file format (see section 6: *File formats*)
 - Standards compliance (see section 7: *Standards support*)
 - Kinds and levels of support and training required by the tool
 - Collaborative authoring (vs standalone authoring)
 - Number, roles, and distribution of potential tool users
 - Bandwidth and other IT constraints and opportunities
 - Budget for purchasing tool(s) and associated support/training contracts
2. Determine category(ies) of tools you will need (see section 4: *Categories of authoring tools and examples*).
 3. Identify specific tools for the identified key category(ies) (see section 4: *Categories of authoring tools and examples* for example tools in each category).
 4. Develop and complete a matrix that allows assessing the tools identified in step 3 against your requirements developed in step 1. See the Appendix for a sample. You may want to complete a separate matrix for each different category of tools you have identified as a requirement for your organization, since each category of tools has its own distinct parameters and typical feature sets.
 5. Filter the list of potential candidates, eliminating those that do not meet your minimum requirements.
 6. Compile a detailed and complete features list for all of the remaining candidate tools. You may want to develop this list from sampling one tool that seems to be the most feature-rich, and build the matrix out as you complete the comparison process. You may want to edit this list of features to only those that you care about now; however, we do not recommend this since you may be unfamiliar with the usefulness of some features and they may be useful in the future.
 7. Develop and complete a matrix using this list to compare with the list of tools. Assign a numerical rating for each cell in the matrix, indicating degree of implementation of that feature (which could be 0 if it does not have that feature). The matrix should weight each feature, enabling a rollup score for each tool. Complete as much of this matrix as possible from the tools' marketing documentation. See the Appendix for a sample.
 8. Contact tool vendors and ask for detailed information (a live presentation is ideal). Complete the matrix with the additional information.
 9. Make your decision based on feature comparison (weighting each for relative importance), taking into account price and any intangibles.

4. Categories and examples of authoring tools

Authoring tools run a wide gamut. This section outlines the major categories and subcategories of available tools. These categories are key to choosing an authoring tool, since they set the stage for

allowing you to align your e-learning product requirements to tool types and characteristics. It is important to note that these categories are not mutually exclusive. Many tools have elements that qualify them for two or more categories. However, most tools can be assigned to one category as its primary intended use or design architecture.

The following is an outline and description of the types of authoring tools, with examples. The web sites listed for each provide feature sets and further details on each tool. Note that some tools appear in more than one category, as they fulfill multiple purposes.

Note: the lists of examples are not comprehensive, nor do they represent an endorsement of particular products. They are based on ADL's knowledge and research conducted as of 12/23/08. Section 11: For additional Reference lists, visit web sites that may provide more comprehensive and updated information about specific tools.

4.1 Self-contained authoring environments

These applications enable building entire e-learning courses using capabilities within the authoring tool; they do not rely on externally created documents (except for media assets and possibly databases). These generally incorporate WYSIWYG features for screen layout and design, and use an object-oriented approach for structuring course elements and activities. All of the tools in this section are desktop-based applications, except for those listed in section 4.1.3.1: *Web-based e-learning development tools*).

4.1.1 Web development tools

These are open-ended tools for web page/site design; they can be used for any type of web-page/site, including e-learning. Once your organization has developed templates and established workflows, these open-ended tools can work well for authoring e-learning. All create output in standard e-learning web formats such as DHTML. Examples are:

- Dreamweaver®
<http://www.adobe.com/products/dreamweaver/>
- Expression Web®
<http://www.microsoft.com/expression/products/Overview.aspx?key=web>

4.1.2 Rapid Application Development (RAD) tools

These are open-ended tools for designing robust interactive applications (usually for web delivery). They produce binary run time files that are executed by a player or plug-in. Examples include:

- Expression Studio®
<http://www.microsoft.com/expression/products/Overview.aspx?key=studio>
- Flash®
<http://www.adobe.com/products/flash/>
- Flex®
<http://www.adobe.com/products/flex/>

4.1.3 E-learning development tools

These tools are specifically designed to produce e-learning, generally in one or two output file format options. These systems are what training professionals are most commonly referring to when using the term “authoring tools.” The system architecture often relies heavily on templates and “skins” to maximize

production efficiencies; in some cases, the developer cannot create templates, the vendors must create them.

4.1.3.1 Web-based e-learning development tools

These tools are web-based in terms of the authoring tool itself, not just the output files. These server-based applications have the advantage of enabling collaborative authoring and permission/role-based production workflows. Some web-based applications require installation of a thin desktop client. The web-based aspect also enables centralized control and enforcement of standards. Examples include:

- Authoring Instructional Materials (AIM) [government owned]
<http://nawctsd.navy.mil/Programs/TrainerDescriptions/UnderseaPrograms/AIM.cfm>
- Content Point®
<http://www.atlantic-link.co.uk/>
- Course Avenue®
<http://www.courseavenue.com>
- Firefly Publisher®
<http://www.mzinga.com>
- Force Ten®
<http://www.outstart.com>
- Krues [government owned]
<http://www.lsjjax.com/prodserv/tss/webtools/kreus.htm>
- Luminosity®
<http://www.cm-luminosity.co.uk/>
- Mohive®
<http://www.mohive.com/index.asp?cls=product>
- RapideL
<http://www.rapidel.com/rapideli.html>

4.1.3.2 Desktop-based e-learning development tools

Many vendors are moving away from desktop-based authoring applications since they cannot be used collaboratively; some are retaining desktop-based versions as an option. Desktop-based applications generally perform better than their web-based cousins, and have more features. Some desktop tools (for example, video editing tools) do not have web counterparts due to high minimum performance requirements.

Examples include:

- Captivate®
<http://www.adobe.com/products/captivate/>
- Desktop Development Suite®
<http://www.outstart.com/desktop-development-suite-overview.htm>
- E-learning Suite
<http://www.adobe.com/products/elearningsuite/>
- eXe
<http://exelearning.org/>

- Learning Content Development System (LCDS)
<http://www.microsoft.com/learning/tools/lcds/default.mspx>

4.1.4 Simulation development tools

These tools are specifically designed for developing simulations and their component animations. Some incorporate scientific data sets that allow modeling of physical phenomena to simulate the real world as closely as possible (for example, weather conditions in a flight simulator). Many RAD tools can create simulations as well.

4.1.4.1 System simulation development tools

These tools are optimized for systems training, allowing easy capture and captioning of interface features with voiceover narration, etc. Examples include:

- Captivate®
<http://www.adobe.com/products/captivate/>
- Capture Point®
http://www.atlantic-link.co.uk/home_capturepoint.htm

4.1.4.2 3D simulation development tools

These tools are used to create 3D animation and to build 3D models with physical world interactions based on scientific data sets. Examples include:

- ESP®
<http://www.microsoft.com/esp/>
- Flex®
<http://www.adobe.com/products/flex/>

4.1.5 Game development environments

Although you can use many RAD and simulation tools to create game-based learning applications; tools in this category are specific to a particular game engine or game standard. Examples include:

- GameStudio®
<http://www.3dgamestudio.com/>
- Torque Game Engine®
<http://www.garagegames.com/>
- Truevision 3D®
<http://www.truevision3d.com/page-14-create-3d-game-development>

4.1.6 Virtual world development environments

Although you can use many RAD, simulation, and game development tools to create virtual world learning applications, these refer to those that are specific to a particular virtual world or virtual world type. Examples include:

- Vizard Virtual Reality Toolkit®
http://www.aesthetic.com/home_frame/home_frame.htm

- World Visions®
http://www.aesthetic.com/home_frame/home_frame.htm

4.1.7 Database-delivered web application systems

These tools represent the ultimate leveraging of the concept of separation of content and appearance; developers store the content (text and media assets) in a database, and apply formats to them on a presentation layer at runtime. This can be a great advantage if learning content information is volatile; you can update content simply and cleanly by replacing objects in a database through a web form. This approach can minimize course maintenance costs for clients by allowing them to make minor updates themselves rather than paying the content developer for every change.

The authoring tools rely on manipulating screen placeholders (that call objects in from the database), and provide form-based methods for configuring and populating the database. These tools require server software to deliver the e-learning. Examples include:

- Cold Fusion®
<http://www.adobe.com/products/coldfusion/>

4.2 Learning content management systems (LCMSs)

These applications integrate the authoring functions with content management, storage, and delivery, leveraging the advantages of integrating these functions. They also generally assemble and deliver the e-learning dynamically at runtime from a central content repository. This provides great flexibility for reuse of content and media. Users do not develop actual files during the authoring process; they assemble virtual learning objects from database and file elements, similar to the database-delivered web application systems described in section 4.1.7: *Database-delivered web application systems*. Examples include:

- Evolution®
<http://www.outstart.com>
- Learn eXact®
http://www.giuntlabs.com/learn_eXact/index.php
- Mindflash®
<http://www.mindflash.com/product.aspx>

4.3 Virtual classroom systems

Vendors design these applications specifically to create e-learning that is delivered via an online collaboration tool (usually one that is optimized for e-learning, with familiar classroom metaphors). Developers use these systems to author synchronous or asynchronous virtual classroom training; most are capable of creating asynchronous e-learning only by virtue of the fact that the synchronous session can be recorded and played back for self-paced learning. These are not standalone systems, because they require files to be generated externally and imported (for example, PowerPoint slides). Examples include:

- Blackboard Learning System®
http://www.blackboard.com/products/academic_suite/learning_system/index
- Centra®
<http://www.saba.com>
- Connect®
[http://www.adobe.com/products/acrobatconnectpro/Social Learning Suite®](http://www.adobe.com/products/acrobatconnectpro/Social_Learning_Suite®)
<http://www.mzinga.com/a/pdf/MzingaDS-HRSolutions.pdf>

- Wimba Classroom®
http://www.wimba.com/products/wimba_classroom/

4.4 External document converter/optimizer tools

These applications usually offer limited ability to develop e-learning from scratch; they are primarily designed to import and convert external documents (usually PowerPoint® and Word® documents) to web-based e-learning (in DHTML or Flash format usually). Often these external documents are legacy ILT (instructor-led training) files (student guides and presentation slides, for example) that need to be converted to e-learning.

Generally, these tools, sometimes called Rapid E-learning tools, are simpler to use than standalone e-learning optimized tools, since they enable developers to use a familiar program like PowerPoint or Word for authoring. Developers then convert these documents to e-learning within the authoring tool, adding interactive features. These tools enable the rank and file user (in many organizations, subject matter experts who are not instructional designers or course developers) to create local training modules. Rapid e-learning tools facilitate rapid development, although usually with a loss of interactivity, and media richness, possibly reducing learning effectiveness).

Critics of these tools note that they often produce courses with a cookie-cutter look and feel, due to the template-driven architecture and technical inexperience of course developers. The fact that developers use Word or PowerPoint as a starting point can also be a limitation for producing interactive, media-rich e-learning, unless the tool provides a substantial ability to add interactions after conversion.

4.4.1 Web-based external document converter/optimizer tools

These offer the same advantages over desktop applications that e-learning development tools provide: collaborative authoring and centralized control/enforcement of standards. The collaborative authoring features are usually less important in this case, however, since these tools are generally simpler and easier to use, thus enabling non-technical staff to do the authoring without requiring developers with specialized skills. Examples include:

- AuthorPoint®
<http://www.authorgen.com/>

4.4.2 Desktop based external document converter/optimizer tools

Examples of these applications include:

- Elicitus Suite®
<http://www.elicitus.com/>
- Lectora Publisher®
<http://www.trivantis.com/products/lectora.html>
- Metamorphosis®
<http://www.easyauthoring.com/>
- Presenter®
<http://www.articulate.com>
- Wimba Create®
<http://www.wimba.com/products/wimbacreate/>

4.5 Auxiliary tools

4.5.1 E-learning assemblers/packagers

These tools assemble objects authored in other tools into an organization/sequence of learning objects, usually in compliance with SCORM standards. Examples include:

- Frameworker for SCORM®
<http://www.i-a-i.com/view.asp?tid=87>
- RELOAD Editor®
<http://www.reload.ac.uk/>
- SCORM Developer's Toolkit®
<http://www.e-learningconsulting.com/products/SCORM-source-code.html>
- SCORM Driver®
<http://www.scorm.com/products/scormdriver.aspx>
- Tenereo®
<http://www.eduworks.com/>

4.5.2 Specific interaction object creation tools

These are generally stand-alone or accessory application modules, often sold either individually or in application suites; vendors design each module to produce a specific interaction. You purchase modules to meet specific interactions needs that are impossible or difficult to create in your primary tool; you may use these tools to create the interactions, and assemble/integrate the code objects into your e-learning course in the primary authoring tool. Examples include:

- Engage®
<http://www.articulate.com>
- Raptivity®
<http://www.raptivity.com/raptivity-software.html>

4.5.3 Media asset production tools

These tools create graphics, audio, video, and animation files. Examples include:

- Final Cut Studio®
<http://www.apple.com/finalcutstudio/>
- Flash®
<http://www.adobe.com/products/flash/>
- Illustrator®
<http://www.adobe.com/products/illustrator/>
- Logic Studio®
<http://www.apple.com/logicstudio/>
- Photoshop®
<http://www.adobe.com/products/photoshop/family/>

4.5.4 Word processors, page layout, and document format tools

These tools create e-learning reference documents and store them in a convenient format (for example, PDF) that preserves their appearance. Examples include:

- Acrobat®
<http://www.adobe.com/products/acrobat/>
- Office®
<http://www.microsoft.com/>
- QuarkXPress®
<http://www.quark.com/>

4.5.5 Database applications

These applications create and configure databases that may be accessed by an e-learning application. Examples include:

- Access®
<http://office.microsoft.com/en-us/access/FX100487571033.aspx>
- Oracle®
<http://www.oracle.com/index.html>

4.5.6 Web-based collaboration and web 2.0 authoring tools

These applications create collaboration mechanisms and peer-to-peer communication functions such as blogs, chats, wikis, and threaded discussion. Use of these features in e-learning is increasing rapidly; some vendors now specifically tailor these tools to support e-learning and their authoring and delivery systems. There is considerable overlap between these tools and the virtual classroom authoring tools category. Examples include:

- Live Meeting®
<http://office.microsoft.com/en-us/livemeeting/default.aspx>
- Participate®
<http://www.outstart.com>
- WebEx®
<http://www.webex.com>

4.5.7 Web page editors

These applications create web pages/sites that may be references or are otherwise ancillary to the main e-learning course screens. The list of tools in this category is the same as the web development tools listed in the self-contained authoring environments category and is included in the auxiliary tools category, since they can also be used to create web pages that are ancillary to the e-learning.

- Easy WebContent®
<http://easywebcontent.com>
- Editor®
<http://www.mozilla.org/editor/>

5. Templates and skins

One of the most dramatic things you can do to streamline your workflow and reduce level of effort (LOE) is to use templates and skins. You may hear the terms “skins” and “templates” used interchangeably, but they differ in that skins are generally style sheets that globally control the appearance and format of screens. They may include complete interface designs that are applied (sometimes dynamically) to basic content layouts, providing most of the formatting for items appearing within the interface as well as the interface itself.

Templates supply a convenient starting point for developing a screen (often including the interface); you simply replace placeholder text, graphics, etc. Skins or templates can include not just visual elements but a large proportion of the functionality of the screen, often based on instructional activities or kinds of interactions.

During the design phase of a project, authors may either develop their own skins and templates or choose them from a template or skin library (usually packaged with the authoring tool.) Instructional designers or SMEs simply choose the template or skin that applies to a screen they wish to build and populate the content. In this way, they can build screens without the need for skilled developer intervention. This saves huge amounts of time, reduces the requirement for technical expertise, and simplifies the authoring process, since authors are just populating a template rather than engineering the whole screen.

Skins can enable local variations on parent content objects providing each organization or learner community with its own visual interface or style for the same base content, managed on the level of a single master copy. Managers can mandate use of templates and skins to enforce uniform standards for e-learning across an organization.

Of course, use of templates can restrict creativity and create e-learning that is “cookie cutter” in look and feel; developers can mitigate this by simply populating a template/skins library with a large number of appropriate designs and screen types.

6. File formats

6.1 Input

Sometimes there is a need to convert materials from ILT to e-learning. The input format in this case is often Word and PowerPoint files. Rather than starting from scratch to recreate these in the output medium, you can use an external document converter/optimizer tool (described in section 4: *Categories of authoring tools and examples*) to automate the conversion into e-learning. The first step is to convert the PowerPoint and Word documents to web pages residing within an e-learning interface; developers can then build e-learning interactivity into these pages (some interactive features may be automatically converted from PowerPoint as well).

If you are in this situation, you need to 1) limit your tool selection to this specific kind of tool 2) carefully assess your input formats to confirm that they match the tools’ support input formats. Do not assume that your legacy ILT documents are in Word and PowerPoint; some may be done in desktop publishing applications like Quark, or may only be available in PDF format (and the originals may be lost!).

6.2 Output

Probably the single most important question to ask when choosing an authoring tool is: “What output file format(s) does it produce”? It is important that you establish your output format before beginning to choose authoring tools. This serves to filter and focus your search considerably, and ensures that 1) the

files will work within your IT and training delivery infrastructure; 2) you are not stuck with a proprietary format that may disappear from the marketplace and eventually leave you with no ability to open and edit the files, nor with the ability to play them in a browser. Countless organizations that used Authorware® as their authoring tool now find themselves in this situation.

Many LCMSs and some web-based authoring tools are designed to assemble and deliver e-learning dynamically, at runtime. They can output complete e-learning files for use on another platform as a convenience, but are generally not used this way. However, even though runtime output files may not be stored on the server, it is important to consider the file format of the files that are delivered to users at runtime relative to factors such as browser compatibility.

Frequently, the type of training you are creating drives the output format. The most important division is between synchronous vs asynchronous learning. Authoring systems differ markedly in their optimization for these types of learning. For instance, for synchronous e-learning, an external document converter/optimizer tool is probably your best choice. Most of these tools offer outputs for synchronous learning such as speaker notes, handouts, or student guides, in Word or PowerPoint format.

The choice of an output format depends primarily on the requirements of your delivery system, as well as on the type of learning that the file format supports. For instance, Flash .swf format robustly supports high levels of interactivity and is supported by most delivery systems when embedded in web pages. However, the range of tools that can natively edit Flash files is much narrower than DHTML.

You will need to check with your IT department to verify the compatibility of desired output formats with the network and firewalls. For instance, some firewalls block Java applets due to potential security risks.

Some output formats (such as Flash) have the advantage of built-in compression and streaming of files at run time. This can make a significant difference in cases where bandwidth is limited.

It is important that you determine whether the files that an authoring tool produces (in a standard non-proprietary format) are 100% editable in other tools that can handle that format, especially those that generate that format natively. For example, some authoring tools that allowed output as source .fla Flash files were not fully editable as native Flash files, although they could be opened in Flash, severely limiting editability. Authoring tools that produce DHTML implement interactivity in proprietary ways (i.e., Java applets) may be difficult to understand, troubleshoot, and edit in another DHTML editor. You may need to take a detailed look at a product's "support" of an output format. The term "supports" may not have the same connotations as the term "is optimized for" or "is built for".

Output formats are becoming even more important as we enter the mobile learning era. Authoring tools have features that support producing files that can play on mobile devices. In the past, developers had to customize e-design e-learning for mobile devices, but that is less often the case with devices like the iPhone.

7. Standards support

7.1 SCORM

ADL has identified the following high-level attributes for all distributed learning environments.

- **Interoperability:** the ability to take instructional components developed in one system and use them in another system.
- **Accessibility:** the ability to locate and access instructional components from multiple locations and deliver them to other locations.

- **Reusability:** the ability to use instructional components in multiple applications, courses and contexts.
- **Durability:** the ability to withstand technology changes over time without costly redesign, reconfiguration or recoding.

To achieve these attributes in distributed learning environments, ADL promotes the use of the Sharable Content Object Reference Model (SCORM). SCORM defines the interrelationship of course components, data models, and protocols so that learning content “objects” are sharable across systems that conform with the same model.

For more information on SCORM, see <http://www.adlnet.gov/scorm/index.aspx>.

It is important to understand that SCORM neither dictates nor precludes any instructional, performance support or evaluation strategy. SCORM does enable object-based approaches to the development and presentation of e-learning.

ADL encourages object-based learning design. This is enabled by aggregating learning content composed from relatively small, reusable content objects to form meaningful units of instruction. Individual content objects can thus be designed for reuse in multiple contexts, and aggregated variously to assemble new components and programs of instruction.

This object-based approach, intended to support reuse, means that content objects must not determine by themselves how to sequence/navigate aggregations that represent parcels of instruction. Doing so would require content objects to contain information about other content objects, which would inhibit their reusability.

To support reuse, SCORM uses metadata to enable content objects to be discoverable through and across enterprises, within distributed content repositories.

This implies that for an authoring tool to robustly support SCORM, the tool must:

- Support object-based learning design
- Allow defining of SCOs at any level of organization
- Include a SCORM metadata editor
- Create SCORM course packages that include all necessary information for the course delivery system (LMS/LCMS/CMS) to properly deliver the course at runtime.
- Allow definition of sequencing and navigation rules for the course organization

Currently, support for SCORM 2004 sequencing and navigation is rare among authoring systems. You may need to accomplish SCORM 2004 sequencing either by directly coding the manifest file (using XML and Javascript) or by using the Reload Editor (<http://www.reload.co.uk>). The Reload Editor provides a GUI interface for creating SCORM packages and for defining sequencing and navigation of the SCOs. However, you should note that the terms and techniques needed to use the Reload Editor require a thorough understanding of sequencing and navigation concepts and logic under SCORM 2004. It is not for the “technically challenged.”

Authoring tools support SCORM features described above to varying degrees. As part of your decision process, it is important to evaluate how fully the tools support each of these. The depth of support for the standard can make a big difference in the LOE to produce compliant e-learning.

Before you evaluate the authoring tools in terms of SCORM compliance, you should determine the target SCORM compliance level (for example, SCORM 2004 3rd Edition). This depends on the compliance level of your course delivery system. Course delivery systems can lag several versions behind the current

level, and since SCORM levels are not all backward compatible, it is important to determine the level of compliance of your course delivery system (and whether it is certified at that level).

NOTE: We highly recommend that you acquire a sample SCORM-compliant e-learning course produced by the tool you are evaluating, and test it on your target course delivery system. Course delivery systems implement the same SCORM compliance level differently in some cases; the interaction of the particular implementation of SCORM in the course delivery system and the particular implementation of SCORM in your SCORM course package, even if both are at the same level of compliance, may uncover issues. This may impact your decision to purchase a particular tool.

You should also run a sample SCO produced by the authoring tool through the SCORM Compliance Test Suite (download from <http://www.adlnet.gov/scorm/20043ED/cts.aspx>) to verify the level of compliance the authoring tool achieves. The current version of SCORM is 2004 3rd Edition.

There is a SCORM Adopter listing (that includes authoring tools) on the ADL public web site. To see a list of authoring tools, go to <http://www.adlnet.gov/>, click **SCORM®** in the top menu bar, then **SCORM Adopters** in the right sidebar. Choose **Authoring Tool** in the Category drop-down list, and select other options as needed to see a filtered list. Alternatively, there is a complete unfiltered list of SCORM adopters at

http://www.adlnet.gov/scorm/adopters/product_search.aspx?v=0&c=3&s=0&k=

These tools, along with many of the those listed in section 4: *Categories of authoring tools and examples* of this document, have built-in features to support achieving SCORM compliance; in most cases, however, some manual coding is necessary to create fully SCORM-compliant e-learning (for example, in the HTML/JavaScript “wrapper” for SCOs). Furthermore, many of these tools do not automate the creation of SCORM course packages (.zip files containing XML manifest files that describe SCOs, metadata, etc.). The Reload Editor provides that capability.

7.2 Section 508

If your organization requires Section 508 compliance for e-learning products, it is critical that you include this as a decision parameter in your choice of an authoring tool. You can reduce the LOE in developing e-learning substantially if your authoring tool(s) has built-in 508 compliance support, so that there is no need to undertake additional production steps (especially highly technical ones) outside the normal course of the authoring process done within the authoring tool. Your authoring tool should not only add elements to produce compliant e-learning code as you work with it, it should also prevent you from doing things that would produce non-compliant code.

A built-in compliance checker within the authoring tool can be useful, but you should also verify compliance by testing with screen reader software used by those with visual impairments and/or using an independent accessibility checker (see

http://www.rnib.org.uk/xpedio/groups/public/documents/PublicWebsite/public_tools.hcsp)

For a complete summary of considerations related to authoring tools and Section 508, see the WC3's *Selecting and Using Authoring Tools for Web Accessibility* at

<http://www.w3.org/WAI/impl/software.html>

The WC3 also publishes an *Authoring Tool Accessibility Guidelines* document at

<http://www.w3.org/TR/ATAG10/>

For references and other information on Section 508 compliance, see <http://www.section508.gov/>

8. Criteria for assessing quality and suitability of tools

The following are general characteristics of robust authoring tools, irrespective of category as defined in section 4: *Categories of authoring tools and examples*. Some of these criteria may not apply to your situation. A high quality tool:

- Allows use of a wide variety of instructional strategies and learning types.
- Is easy to learn and use, ideally with the ability for users to choose from tiers of features according to the knowledge and expertise of the user. This allows users to start using the program quickly and gradually progress to more complex authoring tiers/feature sets as their skills mature. In other words, users only see features that are relevant to their level of skill and the kind of operations they are capable of performing.
- Provides user interface customization (not on the level of tiers of features, as above, but on an individual feature basis), so that users can optimize for their particular needs.
- Provides a high level of support for standards such as SCORM (3rd Edition is the current standard), Section 508, and AICC, among others. Section 7: *Standards support* describes the criteria for SCORM and Section 508. Does the tool include compliance checkers?
- Supports many media file formats (especially formats that are used in your organization), as well as configuration of the media files within the tool. For example, the tool can embed playback parameters and controller interface for a video into the web page. Note that many authoring tools cannot produce simulations requiring complex variable manipulation; you need a specialized tool for that.
- Has options for outputting print files to enable students to print sets of course screens, rather than limiting them to the browser print function (which will only print the current screen).
- Integrates the storyboarding process into the tool. Some tools integrate storyboarding with the authoring process, so that most files can be output when the storyboard is complete, without any need for further production. Some tools support using PowerPoint to create storyboards, which can be imported, edited, and turned into e-learning.
- Provides support for creating student surveys and certificates. Many course delivery systems do this, but you may want this feature in your authoring tool.
- Supports a wide variety of delivery architectures. For instance, if you have an e-learning architecture involving a dedicated content repository (that may be on a different server than the course delivery system), the tool supports configuring the e-learning for this.
- Supports creation of a desktop executable file that can run on CDs or DVDs or run on the desktop after being downloaded from the intranet when there is limited bandwidth.
- Is easy to install and configure, ideally not requiring a system administrator.
- Has reasonable system requirements that are attainable within your organization.
- Manages the production process efficiently. This may include built-in workflows (for approvals, for instance) and production, QA, and review pipelines.
- Offers “organization aware” features that allow collaborative server-based authoring based on organization roles and permissions.
- Is optimized for reusability in general (not just measured by SCORM support). Some tools have their own internal content repository that allows mixing and matching objects.

- Has a licensing agreement that is flexible and easily scalable to reflect changing number of authors.
- Includes a wide variety and numbers of templates or skins that you can use out of the box with little or no modification. This can save time and simplify development greatly.
- Has the ability to call external applications and code objects (such as calculators and random number generators), and set up interfaces to read and write from databases.
- Provides the ability to develop a wide variety of assessment types (possibly with templates).
- Permits programming sophisticated navigation and sequencing of content objects (ideally using built-in features that enable SCORM 2004 sequencing and navigation).
- Is low cost compared to competing authoring systems with the same or similar feature set.
- Has robust documentation and support: help, examples, references, user manuals, support options, etc.
- Offers extensive training options: e-learning, video tutorials, ILT sessions, webinars, etc.
- Provides the ability to annotate and communicate actions taken, approvals, errors, etc. in regards to screens and content objects, for future reference, or for other authors.
- Includes a variety of interface metaphors for reviewing the content structure, for example, flowcharts and object hierarchies.
- Incorporates some degree of ability to edit raw material assets at a low application level; for example, support for editing of images to the degree that use of Photoshop might not even be necessary.
- Offers a scripting language (such as Adobe Flash ActionScript®).
- Supports output to mobile devices.
- For template-driven tools, supports developer-generated templates.
- Supports a sufficient number and flexibility of assessment types. Most tools have special templates and features for creating assessments. The standard types of e-learning assessments that you should look for are:
 - Multiple choice (both single and multiple answer)
 - Fill in the blank
 - Matching
 - Drag and drop
 - Ranking/Ordering
 - Image selection
 - Essay or Short answer (this requires instructor intervention to score answers)
- Includes robust programming features, such as:
 - Browser emulation that allows quick previewing of screens and objects exactly as they appear and function in the target browser.
 - “Round-trip” editing of code, meaning that changes made while in code editing mode (for example, directly making changes to the HTML code) are immediately reflected in WYSIWYG editing mode, and vice versa.

- Ability to launch source object editing applications from within the tool. For example, the ability to launch and edit graphic objects in Photoshop from within the tool; saving changes automatically updates the file in the target format (like JPEG).
- No special non-standard or proprietary code is used in conjunction with HTML (for example, some authoring tools rely on special code inserted into HTML comments fields).
- Ability to set control parameters for media objects (for example, Flash animations) within the tool rather than requiring them to be set within the media object authoring tool. This includes looping behavior, streaming parameters, etc.
- In LCMS tools, a high degree of traceability for the impact of all changes across all courses and objects. For example, properties and attendant warnings that object x is used in courses y and z; changes to that object will affect those courses, and vice versa.

It is important to remember the simple fact that most users, in many cases regardless of their skill set, will follow the path of least resistance in using an authoring tool, as with any other software. In other words, users will gravitate towards the most heavily optimized system features—those that are prominently available in the interface and easiest to manipulate. The system may include many advanced capabilities, or even easy workarounds or hacks that are possible, but most users will ignore these. This can drive a “dumbing down” of the sophistication and quality of the end product, reducing learning effectiveness to below desirable levels.

So the question is not necessarily, “What can the software do?” but, “What can the software do in a right-out-of-the-box, plug-and-play use case scenario?”

It is important to determine the skill sets within your pool of course authoring staff, so that you know what you are prepared for and/or what you might have to acquire in terms of staffing or training.

Another important point to keep in mind is that tools that are easier to learn and use have fewer capabilities, and vice versa. Sophisticated media and/or learning strategies will inherently require a tool that is harder to learn and/or require specialized professional expertise.

9. General recommendations

- Avoid the first release of a new authoring tool.
- Ask the vendor who their other clients are, what they use the tool for, and see if you can talk to these clients about their experience using the tool.
- Ask the vendor for a demonstration in your facility, running on your enterprise system(s). The vendor may present a canned demo of the product on their system, and that is fine as a general overview of the tool’s capabilities, but you should see how well the tool expresses these capabilities within your IT environment.
- Avoid authoring tools created by companies that have a short history in the market, or have been operating for a short time, or are small.
- Determine exactly what capabilities you really need. If you already have a course delivery system, for instance, you may not need that capability that is included in an LCMS. Many LCMS vendors sell the authoring module as a separate application for a lower price.
- For a web-based tool, determine whether a hosted solution may be right for you. In most cases, outright ownership is the best route. However, a hosted solution may be cheaper in the long run,

in terms of server usage and saving the hassle of performing your own admin and maintenance functions.

- Consider using several authoring tools in combination; a primary one for authoring the “shell”, and auxiliary authoring tools that are optimized for particular capabilities or assets. This is very commonly done in the case of courses that are authored in DHTML (for instance, using Adobe Dreamweaver), with Adobe Flash objects inserted for animations.
- Try the tool out on the system configuration your authors typically would use in your training organization. You may discover some surprises in performance and features that you would not otherwise have found. For instance, the authoring tool’s Preview function may actually preview screens quite differently than what they look like in the actual end-user browser.

10. Current Trends in Authoring Tools

10.1 Team-based life cycle production and maintenance

There is a recent trend towards support for team-based life cycle production and maintenance. This includes all of the phases of an e-learning project: analysis, design, development, implementation, and evaluation (ADDIE). This trend is driving many desktop tools to move permanently to web-based architecture, or at least to have a web-based option, since this enables all kinds of "organization aware" workflows (enforcing who does what when).

10.2 XML Output

Like the software area in general, a lot of tools are moving towards some version of XML, as output and/or as the internal authoring language (perhaps not on the presentation layer of the application, but on the back end). XML is becoming a universal language for many purposes. Because of its perceived durability and stability, some training organizations are requiring that their learning content be in this format.

10.3 Separation of content and appearance

For quite a while now, there has been a trend towards separation of content and appearance. Examples of this are tools that use technologies like Cold Fusion and server technologies like ASP. This trend has permeated deeper and deeper into application architecture. Separation of content and appearance facilitates flexible updating of text, media files, etc. without recoding the screens they appear on. Some tools that use this principle rely on dynamic assembly of e-learning at run time (which requires server software); others assemble and solidify the final product at the time files are published (handled within the authoring tool). Another way that separation of content and appearance manifests is the separation of the course interface from the content. Most often this involves use of skins, which are interface designs (possibly including functionality as well as visual design) that can be swapped out easily.

10.4 Centralized control with distributed contribution

This trend is evident in the many web-based tools that allow collaborative authoring and permission-driven production pipelines. More and more, tools allow modeling of organization structures and processes, and assignment of specific roles in the production process.

10.5 Templates and skins

This was discussed in section 5: *Templates and skins*. Templates and skins have always been a part of authoring tools, but they are becoming much more integral to the tools, and are becoming more complex.

10.6 Learning object-centric architecture

Authoring systems that are integrated with Learning Content Management Systems (LCMSs) or content repositories best exemplify this principle. They give developers the flexibility to develop all kinds of content objects (not just explicitly designed for learning purposes) and assemble and reassemble them in different combinations (often relying on SCORM to do this) for learning modules either at runtime or when courses are published. This trend reflects the growing popularity and movement towards knowledge management practices.

10.7 Embedded best practice design principles

Tools are integrating visual and instructional design principles more and more, as these principles are more accepted and standardized, and become the default working principles for e-learning.

10.8 Support for advanced learning technologies

Finally, there is growing interest in game-based learning, intelligent tutoring systems, and virtual worlds. Tools are now catching up to support these.

11. For Further Reference

- *Creating Instructional Multimedia Solutions: Practical Guidelines for the Real World* (2005, Peter Fenich). This book contains information about delivery
- *Directory of Learning Tools* (Centre for Learning and Performance Technologies) <http://www.c4lpt.co.uk/Directory/Tools/authoring.html> (accessed January 13, 2009). This web site contains a detailed list of available authoring tools, with review comments.
- *Rapid E-learning Authoring Tools* (Kineo Corporation). <http://www.kineo.com/authoring-tools/rapid-e-learning-authoring-tools.html> (accessed January 13, 2009). This web site contains a report on a survey of users of authoring tools.
- *Course and Lesson Authoring Tools* (E-learning Centre (UK)) <http://www.e-learningcentre.co.uk/eclipse/vendors/authoring.htm> (accessed January 13, 2009). This web site contains a detailed list of available authoring tools with abstracts describing each.

Appendix

Sample Tool Requirements Matrix

The following is a sample of a matrix that can be used in step 4 presented in section 3: *Process for choosing tool(s)*. The step is described as:

Develop and complete a matrix that allows assessing the tools identified in step 3 against your requirements developed in step 1.

To use the matrix:

1. Replace the top row with items you have determined to be your requirements for the tool. For example, for “Standards compliance”, you could substitute “SCORM 2004 3rd Edition, Section 508”.
2. Put the list of tools in the “Tool name” column.
3. Research and complete the cells with information indicating whether each tool meets that requirement (may be “yes” or “no”, or a more lengthy description of how it meets or doesn’t meet the requirement).

<i>Tool Requirements Matrix</i>											
	Media supported	Interactivity level	Authoring skill set	Editing by non-technical staff	Output file formats	Standards compliance	Support and training req'd	Collaborative authoring	Number, roles, and distribution of users	IT considerations	Budget
Tool name											

Sample Tool Features Rating Matrix

The following is a sample of a matrix that can be used in step 7 presented in section 3: *Process for choosing tools*. The step is described as:

Develop and complete a matrix using this list to compare with the list of tools. Assign a numerical rating for each cell in the matrix, indicating degree of implementation of that feature (which could be 0 if it does not have that feature). The matrix should weight each feature, enabling a rollup score for each tool. Complete as much of this matrix as possible from the tools’ marketing documentation.

To use the matrix:

1. Replace the top row (Feature1, Feature 2, etc.) with the names of all relevant features you have compiled for the authoring tools.
2. For each Weighting factor cell in the row below it, replace the text with a number between 1-3 to weight the relative importance of that feature (the higher the number, the more important).
3. Put the list of tools in the “Tool name” column, then research the feature information for each tool and complete the cells with the number indicating the degree to which each tool has that feature (we suggest 0-2, 0 being “does not have that feature” and 2 being “has implemented this feature to the fullest extent possible”).
4. The rollup score column at the far right will provide the total weighted score for each tool (right-click on it and select **Update Field** after you make any changes to the weighting values or ratings).
5. If you add columns or rows, copy and paste the Rollup score formula and adjust the row and column references in the formula accordingly. Right-click the pasted Rollup score and select **Toggle Field Codes** to see and edit the formula.

<i>Tool Features Rating Matrix</i>											
	Feature 1	Feature 2	Feature 3	Feature 4	Feature 5	Feature 6	Feature 7	Feature 8	Feature 9	Feature 10	Rollup score
Tool name	<i>Weighting factor</i>										
											0
											0
											0
											0
											0
											0