Encouraging Reuse at the Source:
Use Cases for Creating Metadata and Facilitating Content Management within a Development Environment

The Academic ADL Co-Lab
with support from
The William and Flora Hewlett Foundation
Academic ADL Co-Lab Staff

Table of Contents

Abstract 2
Overview 2
Introduction 2
Developers and Development Environments 4
What is a Use Case? 5
Applying Use Cases 6
    Making Metadata Production Possible at any Point within the Production Process 6
    Exploiting the Development Environment to Automate Metadata Creation 10
    Versioning without Metadata 12
Concluding Remarks 14

Abstract

A successful reusable content development strategy would encourage developers to submit to repositories standards-conformant metadata with the actual content they create. This might be achieved by integrating metadata creation into the development process. To facilitate this integration, developers should be able to create metadata at any point in the production process. Additionally, some level of automatic metadata generation should be developed. Finally, versioning information could be created and used independently of metadata. All of these functionalities can be facilitated by the development environment and repository.

Overview

Through a grant from the William and Flora Hewlett Foundation, the Academic ADL Co-Lab is in the process of evaluating and developing tools for the demonstration of repository functionalities. This project is continuing with the Co-Lab's on-going commitment to applied research into strategies that encourage the sharing of online educational content.

Before developing the actual tools for metadata creation, we have developed use cases which outline the necessary sequence of events that a developer would go through to be able to easily create metadata alongside her content, either manually or automatically. These use cases define the specific prerequisites that developers in a variety of environments might encounter

This paper investigates convenient metadata production functionalities that could encourage the reuse of educational content. These functionalities would encourage the aggregation of interoperable metadata by facilitating its generation within a production environment. This paper also proposes that in cases where metadata and content generation are distinct activities in a production environment, data about that content can be encoded prior to the creation of formalized metadata.

Each section of the paper will describe:

- The proposed functionalities,
- Use cases that describe the process of implementing these functionalities, and
- Drawing upon the use cases, the functional requirements that they entail.

Introduction

Online content is becoming a more common part of education.[1] Educators are integrating online content into their teaching; content which they often develop

[1] Sloan Consortium, "Entering the Mainstream: The Quality and Extent of Online Education in the United States, 2003 and 2004" http://www.sloan-c.org/resources/entering_mainstream.pdf

themselves. At the same time, faculty support personnel and developers are creating online educational content for learners and educators to use.

This has raised questions about how this process can become more efficient. As both educators and developers are creating content, they could benefit from resources that encourage reuse.[2] High quality content production can be difficult, and the rewards could be maximized if the content is designed to be shared and used again.

If content were available for reuse, some amount of duplication of effort could be avoided. To make that content available we should put forward some model for sharing and discovering it.

Repositories, online collections of content and metadata about that content, provide a means by which online material can be shared. Repositories that can share metadata, allowing for searches across collections, can permit searchers to discover and reuse a wide variety of educational content.

Placing content within a repository almost always requires the production of metadata. If metadata is to be shared, that metadata must be standards-conformant, allowing it to be shared by multiple repositories and ensuring that the shared metadata can be useful and coherent when aggregated.

Metadata production, however, has proved to be a challenge. The great majority of repository projects has depended upon the public for metadata production.[3] Rules and interfaces guiding users through the multiple steps of metadata production will encourage consistency throughout metadata production workflows.

This paper proposes that we can encourage the creation of standards compliant metadata by drawing upon a very specific group composed of developers and development environments. This paper demonstrates ways in which the production of metadata can be integrated into the production process of creating learning content.

This paper provides use cases to describe scenarios in which a development shop could integrate the production of metadata and other useful data into the development process. The scenarios demonstrate that:

- Metadata production should be possible at any point within the production process,

[2] The drive to create repositories that will encourage reuse has been discussed in the Academic ADL Co-Lab's "From Local Challenges to a Global Community: Learning Repositories and the Global Learning Repositories Summit" http://www.academiccolab.org/resources/FinalSummitReport.pdf

[3] Well known examples of this include repository projects such as MERLOT, the Maricopa Learning Exchange, and some SNDL-related projects such as SMETE. Many metadata creation tools, such as Reload, the Scout Portal Toolkit and CWIS, and the SPLASH interface, are developed to create metadata post item production or location.

- Metadata production can be generated automatically from production processes, and that,
- Useful versioning information can be generated automatically and independently from metadata, during the production process.

Through these use cases we propose the ways in which regular production processes that may already exist can be drawn upon to create similarly regular metadata.

Developers and Development Environments

Online educational content is developed, altered, versioned, shared, and delivered in a number of contexts. As the development of online content becomes a more widely distributed skill, and as the tools that aid in development become widely available, making general statements about who produces online content and where production occurs will become more difficult.

This paper does make an assumption in this domain. We can assume that some amount of production will always be done by persons and projects with a professional or institutional commitment to content development. These persons we will call developers and the projects in which they work development environments.

As web-based content becomes a more common part of education, the content that is demanded by learners and educators will become more complex. At the same time, advocates of learning objects discuss the development and use of online materials with which competencies can be both learned and demonstrated.[4] However, creating materials that meet any of these challenging demands requires time, effort and expertise.

Thus we can assume that some quantity of online educational content will be developed, updated, and edited by the professional developers who can be trusted to have development expertise, and not individual educators creating their own content.

This paper focuses on developers and development environments because we can reasonably assume that they will always have a hand in the production of at least some educational content. We can also assume that the institutional structures within development environments will assure a production process more predictable then what we could find among individual faculty developers or those creating content rapidly to meet specific, immediate needs. It is these institutional structures that we draw on in our scenarios, demonstrating the automatic production of metadata and versioning information.

The Academic ADL Co-Lab has also drawn upon its previous research in focusing on developers. During the spring of 2004 the Co-Lab conducted a series of surveys to

[4] David Wiley "The Instructional Use of Learning Objects – Online Version," http://www.reusability.org/read/

discover what functionalities respondents felt were needed in a repository system.[5] Those surveyed include potential repository users, and persons involved in repository development projects.

Our results showed a general awareness among most respondents that a repository needed to be able to conform with the content development processes already found within education. More importantly for this paper, however, we found an interest in developers' needs. Respondents were very consistently interested in repository use of WebDAV[6] and nearly half of the respondents to our first survey expressed an interest in meeting developer needs. One respondent stated that she saw the "repository being used at the author/developer [level] rather than at the user level."

We believe these results indicate a need to look at how developer activities can be facilitated, and how their activities could be integrated into the functioning of repository systems. The direction of our inquiry arose from the experiences of the Academic ADL Co-Lab's staff both as content developers and providing technical support to developers.

What is a Use Case?

A use case is a detailed description of the preconditions and steps required for a user-system interaction to bring a simple action to completion. The function of a use case is to take the complex processes involved in user-system interactions and reduce them to simple steps that describe the requirements placed upon a system.[7]

Within the use cases, the person(s) or systems interacting with a repository/development system is described as an actor. An actor participates in the process of utilizing a use case. The objectives of these actors will be specified in each use case. A resource is a unit of content. For these use cases we will assume that a resource is any piece of content for which it is appropriate to author metadata.

The use cases in this paper will include the following sections:

- Name: A title for the use case.
- Description: A brief description of how the actor within a use case is interacting with a system.
- Preconditions: Technical or institutional requirements for the process to take place.

---

[5] "What We Mean When We Say 'Repositories' User Expectations of Repository Systems" http://www.academiccolab.org/resources/RepoSurvey2004-1.pdf

[6] WebDAV stands for Web-Based Distributed Authoring and Versioning, and provides a way to remotely author and manage content. WebDAV is a protocol standardized by the IETF

[7] Richard Woolridge "Introduction to Use Case Analysis," http://www.cbd-hq.com/PDFs/cbdhq_991115rw_usecaseanalysis.pdf

- Sequence of Events: The specific steps required to bring the process of actor-system interaction to its expected end point.

Applying Use Cases

Making Metadata Production Possible at any Point within the Production Process

If a group of developers wishes to create metadata for the content they produce, certain difficulties will arise. One of these is determining when to create metadata during the production process.

Many of the repository systems and metadata authoring tools we have encountered have been designed under the assumption that metadata will be produced in its entirety after the content it describes has been created.[8] However, to facilitate the easy creation of metadata developers should be able to generate metadata associated with a specific piece of content at any point in the production process. This would prevent metadata generation from constricting a development environment's workflow.

The following use cases describe multiple scenarios for metadata generation within a production process.

The scenarios we have outlined are the following:
1) Saving Resources
2) Creating Metadata
3) Saving Metadata
4) Saving Resources and Pre-existing Metadata
5) Editing Saved Metadata
6) Saving New Metadata for Existing Resources

Use Case One	
Name	Saving Resources
Description	An actor needs to store newly developed resources.
Preconditions	A method for identification of content is in place.
Sequence of Events	1) The actor initiates a request to save a new resource or resources. 2) An identifier is assigned upon successful submission of the resource. The identifier, and a method for resolution, is made known to the actor and associated with the resource.

Use Case Two

[8] Such tools would include the Gateway to Educational Material's GEMCat, the Scout Portal Toolkit and CWIS, as well as Reload, an educational metadata editor.

Name	Creating Metadata
Description	Actor creates and saves a new metadata record for previously saved resources.
Precondition	A method for identification of content and metadata must exist. A metadata creation interface must exist. Submitted data is, or can be transformed into, standards-conformant metadata.
Sequence of Events	1) Actor specifies the identifier for the saved resource for which he or she wishes to create metadata. 2) Selected resource is located. 3) A metadata submission form is returned. 4) Actor submits data using the metadata creation interface. 5) Submitted data and the identifier of the corresponding resource are stored. 6) An identifier for the new metadata is generated and saved.

Use Case Three	
Name	Saving Metadata
Description	Actor is in possession of standards-compliant metadata for a resource. The resource is already stored. Actor saves this metadata.
Precondition	A method for identification of content and metadata. Metadata for a previously submitted item can be altered while maintaining that metadata's standards compliance.
Sequence of Events	1) Actor specifies the identifier for the saved resource for which he or she wishes to submit metadata. 2) System locates the selected resource 3) Actor is prompted to identify the metadata document to be saved. 4) Metadata is saved. 5) Metadata is altered to include accurate description of the corresponding resource's location and identifier. 6) Metadata and the identifier of the corresponding resource are stored. 7) An identifier is assigned to the new metadata

Use Case Four	
Name	Saving Resources and Pre-existing Metadata
Description	Actor wishes to save a resource and existing metadata corresponding to that resource.
Precondition	A method for identification of content and metadata. Metadata for a previously submitted item can be altered while maintaining that metadata's standards compliance.

Sequence of Events	1) The Actor initiates a request to store a new resource. 2) Actor initiates a request to store corresponding metadata. 3) An identifier is assigned and returned upon successful submission of the resource. 4) Metadata is altered to contain an accurate description of the corresponding resource's location and identifier. 5) Altered metadata is saved. 6) An identifier is assigned to the saved metadata.

Use Case Five	
Name	Editing Metadata
Description	Actor wishes to alter metadata for a resource.
Precondition	Identifiers must be actionable, returning the resources they represent. A metadata editing and creation interface must be provided .
Sequence of Events	1) Actor retrieves metadata for a resource by specifying the resource's identifier 2) Existing metadata is presented within a metadata creation/editing interface. 3) Actor alters and adds to metadata within the creation/editing interface. 4) New metadata is saved.

Use Case Six	
Name	Saving New Metadata for Existing Resources
Description	Actor wishes to save new metadata for a resource which is already saved and for which metadata has already been saved
Precondition	A method for identification of content and metadata. Identifiers must be actionable, returning the resources they represent.
Sequence of Events	1) Actor specifies the resource for which he is uploading metadata. 2) Actor identifies the metadata record to be saved. 3) Resource's existing metadata is located. 4) Existing metadata is returned. 5) Actor initiates save of new metadata. 6) Metadata is altered to contain an accurate description of the corresponding resource's location and identifier. 7) Altered metadata is saved. 8) An identifier is assigned to saved metadata.

Discussion

Developers do not want extra steps imposed upon their production process unless they lead to an overall increase in efficiency. Developers and development shops may invoke any number of production processes involving an unpredictable number of persons, tasks, and stages. Developers will therefore be most served by functionalities that allow metadata to be produced at any stage in the production process.

Above we have described a number of scenarios in which metadata is created, edited, and associated with a resource. Drawing on the use cases we can conclude that a system that is able to carry out all the processes described above must:

- Be able to identify material being saved as a resource or metadata, and initiate distinct processes in each case
- Be able to alter metadata to specify the location of content after it has been saved, and identify the appropriate metadata and content when required
- Generate and utilize identifiers to locate resources and their corresponding metadata

Be able to identify material being saved as a resource or metadata, and initiate distinct processes in each case

We can imagine a system which allows us to create and save metadata at a specific point in the production process, understanding that metadata and content will be saved in unpredictable patterns; sometimes together, sometimes one after the other, and sometimes with long gaps between the saving of one or the other.

To allow this, some means must exist for allowing the system to specify when uploaded resources are metadata and when they are content to which metadata refers. Metadata production can be part of a regular content-production workflow, but a production system should also support the creation of metadata outside of that workflow.

Be able to alter uploaded metadata to specify the location of content after upload and identify the appropriate metadata and content when necessary

The capacity to locate metadata fields within saved metadata, and to correctly assign new values to those metadata fields is absolutely necessary. This requires that the system recognize and be able to alter metadata to be conformant with some set of supported metadata schemas.

Generate and utilize identifiers to locate resources and their corresponding metadata

As long as metadata and resources may be uploaded at distinct points in time, or multiple metadata records may be uploaded to describe a single resource, a system cannot depend entirely upon information within that metadata to specify the identity or location of the resources to which they correspond. For example, we can imagine stages in the production process in which the metadata for a resource has not been created, but a developer still needs to know that resource's location. Thus some system of identifiers as well as a resolution system must exist for both resources and metadata, allowing resources to be recognized without requiring that metadata be referenced.

Exploiting the Development Environment to Automate Metadata Creation

An effective and sustainable strategy to encourage the submission of educational content to repositories is to facilitate the production of standards-conformant metadata by integrating automatic metadata production into content development and management.

Imagine a workshop that is in the process of developing digital educational content. When a resource has been completed the resource and its metadata are to be placed within a repository and made available to searchers. Asking developers to create metadata many not be cost or time effective. When called upon to produce metadata, many users may not see how doing so would directly benefit them.[9] Indeed, the process can take time and mental energy that developers may wish to use elsewhere. This process could be made easier by tools and processes that attach metadata to resources during the production process.

Development environments are able to administer tasks, projects, and workflows with some level of predictability. As long as these remain predictable, automatic metadata generation is made much easier.

Because these use cases assume a development environment, they do not apply to all automatic metadata generation or existing metadata environments. Some of the e-learning content placed within any repository will not come from known development environments, and there will always be a need for multiple forms of automatic metadata production. However, development environments have a place in the production of educational content that should motivate us to conceive of ways of using them to facilitate automatic metadata production.

In these use cases, drawing upon our experiences while developing content within the Academic ADL Co-Lab, we have assumed that in a production environment various persons are assigned to specific projects. These projects produce resources with some consistency as to their subject matter, format, structure, etc.

Within such a structured development environment, a registry of projects could be created which will store the metadata that will be associated with any resources produced by that project. Upon the creation of a resource, a system that used these functionalities could identify the individual author and from that information automatically generate metadata.

The use cases themselves are:

1) Defining a Project
2) Creating Metadata Describing a Project's Output
3) Logging in as a Member of a Specific Project

[9] Sarah Currier "Metadata Quality in e-Learning: Garbage In - Garbage Out?" http://www.cetis.ac.uk/content2/20040402013222 and Barton et al. **"**Building Quality Assurance into Metadata Creation" http://www.siderean.com/dc2003/201_paper60.pdf

4) Automatically Generating and Saving Metadata

Use Case One	
Name	Defining a Project
Description	Actor defines a project within a system and specifies the person(s) who will be considered to be associated with that project.
Precondition	Projects must be defined entities within the repository tool. System must have a method of recognizing users and projects.
Sequence of Events	1) Actor specifies the name of a project. 2) Actor specifies the persons or login names that will be registered as working on that project.

Use Case Two	
Name	Creating Metadata Describing a Project's Output
Description	Actor creates metadata which will be applied to all resources saved by persons logged in to a system as part of that specific project.
Precondition	A metadata authoring interface must be provided. Projects must be defined entities within the repository tool.
Sequence of Events	1) Actor specifies the project for which he or she is creating metadata. 2) A metadata submission form is returned. 3) Actor creates metadata. 4) Metadata is saved for the project.

Use Case Three	
Name	Logging in as a Member of a Specific Project
Description	Actor logs onto a system and is recognized as a member of a specific project
Precondition	User identifiers can be assigned and recognized. Logins are associated with additional data.
Sequence of Events	1) Actor logs in using a registered username. 2) Username is recognized. 3) Data associated with the username is retrieved. 4) Data retrieved includes specific projects to which that user belongs.

Use Case Four	
Name	Automatically Generating and Saving metadata
Description	Actor logs in to a system and saves resources, and some metadata is automatically created.

Precondition	User identifiers can be assigned and recognized. Logins are associated with additional data. A metadata authoring interface must be provided.
Sequence of Events	1) Actor logs in using a registered username. 2) Username is recognized. 3) Data associated with the username is retrieved 4) Data retrieved includes specific projects to which that user belongs. 5) Actor saves resources within a project. 6) Metadata associated with project is retrieved. 7) A metadata authoring interface populated with retrieved metadata is returned. 8) Actor edits metadata in metadata authoring interface. 9) Actor saves metadata.

Discussion

These use cases describe a situation in which an institution has developed mechanisms to ensure some level of automatic metadata production. To be put into practice, the scenarios described require some initial planning on the part of the development project.

The following technical and institutional commitments would also have to be met.

- Institutional structures must exist that can decide upon consistent metadata for a project's products
- Identifiers must exist to refer to specific logins, and associate the logins with pre-existing project information
- The system must be able to generate metadata compliant with a limited set of possible metadata schemas

Versioning without Metadata

In our use cases we have presumed that resources and their metadata are distinct entities, created and saved within distinct workflows. This leaves open the possibility that at some time in the development process a system may contain resources but no corresponding metadata.

Were this the case, functionalities often permitted by the presence of metadata would not necessarily have to be ruled out. Data of importance within a production environment, regarding versioning for example, can be generated automatically in a form that is potentially actionable and useful.

In development environments developers often create multiple versions of a single resource. These versions may be generated with sufficient frequency that developers might search for different versions of a resource before it has had corresponding metadata created. In such a situation some means must exist for searching for and retrieving resources at a point after they have been produced and before metadata has been created to describe them and facilitate searching for them.

Within a production environment, certain processes may take place with sufficient frequency that workflows may be slowed if actors are required to create metadata at multiple points in the production process**. In many production environments developers may create multiple documents in a single day. Creating useful data about these items more quickly and automatically than by enlisting developers in creating metadata themselves is possible.**

The use cases that will be illustrated below are:

1) Creating a version of a resource
2) Retrieving multiple versions of a resource
3) Retrieving a single version of a resource

Use Case One	
Name	Creating a version of a resource
Description	Actor specifies that a saved resource is a later version of a previously saved resource
Precondition	A method of identifying content and metadata A system for relating versions of objects
Sequence of Events	1) Actor initiates saving a resource 2) Actor indicates that the resource is a later version of a resource already saved within the system 3) Actor is prompted to indicate the resource of which the current resource is a version. 4) System encodes relationship between versions in an actionable way

Use Case Two	
Name	Retrieving multiple versions of a resource
Description	Actor wishes to locate all available earlier versions of a resource
Precondition	A method of identifying content and metadata A system for relating versions of objects
Sequence of Events	1) Actor identifies a resource 2) Actor initiates a request for previous versions of the same resource 3) System retrieves all previous versions of the resource

Use Case Three	
Name	Retrieving a single version of a resource
Description	Actor wishes to retrieve a single version of a resource
Precondition	A method of identifying content and metadata A system for relating versions of objects

Sequence of Events	1) Actor designates a resource for which he wishes to locate a version 2) Actor designates the version of the desired resource that they wish to retrieve 3) Resource designated by the user is returned

Concluding Remarks

In the scenarios above we described how developers and development environments could be drawn upon to facilitate the generation of various kinds of metadata. The scenarios show that some technical requirements arise if we are to create tools that can integrate metadata and content production but are "work-flow agnostic" – that is they do not enforce a single, or limited set of, metadata production workflows.

In analyzing these scenarios we have chosen to focus on development for two reasons. First, we believe that the fields of e-learning and educational technology are not made up solely of learners and educators. In the institutional context in which education takes place, there are professional communities of developers and support personnel. Ignoring them is to ignore a community whose efforts are crucial to the advancement of online learning.

Second, we believe that online learning is encouraged not only by individuals, but by institutions. The scenarios we describe above require that development environments meet not only technical requirements, but institutional requirements as well. One could object that this is a potential barrier to their adoption. However, from our own experience developing educational content, we have found that developers have a need for tools that facilitate online development.

These use cases are intended to meet the well-recognized needs of educators and learners for reusable digital educational content. To encourage reuse we should address the needs of those who will accomplish that reuse; developers. Whether connected with instructional materials centers, or grant funded, teams of developers are working with educators to create educational content. These developers can be brought into the metadata creation process, and by developing tools that would address their needs the process would be facilitated.

There are no simple solutions within e-learning. If we are to develop services that meet the needs of educators and learners, we would do well to imagine a set of services, each meeting the needs of groups of constituents already invested in developing and delivering e-learning content. By implementing functionalities that encourage automatic metadata generation and facilitate effective content management, educational institutions can encourage developers to increase the flow of content and standards-conformant metadata into repositories, and ultimately into the hands of the educators and learners who can use them.