A Lightweight Multimedia Web Content Management System

Yong-gang CAO, Nuo LI, Mao-zhong JIN, Yu-qing LAN
Software Engineering Institute, School of Computer Science and Engineering
BeiHang University, China
{cao_yg, seraphic}@sei.buaa.edu.cn {jmz, lanyuqing}@buaa.edu.cn

Abstract

More and more organizations find it more convenient to distribute and collaborate on content through internet than by traditional methods, and the demand for the management and integration of multimedia content among sites is also increasing.

This paper proposes a Lightweight Content Management System (CMS) implementation for multimedia web content. It has a loose-coupled extensible architecture and covers major requirements of a common CMS. It was designed from scratch independent of any existing traditional CMS to adapt distributional environment and emerging world wide collaboration standards. It focuses on establishing skeleton for many additional or enhanced features versus traditional CMS, such as portability, collaboration, supporting multimedia content from multiple sources, extendable meta-data, unified hierarchical taxonomy management, automatic discovery of related content of various media types, taxonomy and role based fine-grained access control etc. We describe the architecture and the design details of the system.*

1. Introduction

Content Management Systems (CMS) manage the content development life cycle from creation and categorization, review and editing, to ranking, live publication, searching content and receiving feedback. Enterprises, government, organizations, and schools all need to share their information and interact with their users through web.

Target on distribution environment, to achieve autonomy and cooperation goal, firstly, we need a CMS with a clearly defined, loose-coupled and service oriented architecture [1]. Based on the architecture, we need establish such a test bed to meet the requirement of reality. A web based multimedia CMS should involve easy managing, fast and secure access in multimedia through network, it also need semantic features for searching and syndicating content. We considered following aspects in addition to a normal CMS requirements:

Portability

A portable system means that it doesn't rely on specific systems. It should be platform-independent, outside system in the same type can be switched without interfering its well working.

Supporting multimedia content, and standard based meta-data

Meta-data is definitional data that provides information about or documentation of other data managed within an application or environment. Content having meta-data is easier to manage exchange and discover than non-Metadata contents.

For exchanging meta-data automatically by applications, the meta-data used must conform to general-accepted standards. We implemented integrated search for all media types based on Dublin Core meta-data standard [2] and defined extendable meta-data data structure as a base for syndication [3].

Accessing content list in different ways

In this paper, we refer taxonomy as widely-accepted classification in specific domain. Applying of taxonomy on CMS allows an organization to unify its terminology and categories. Different users need access content by different ways (by media type or taxonomy).

Showing relationship of content

We need to create relationship between content in the same media type or various media types.

For future support of ontology network [4], we first implemented hierarchy taxonomy management. It is the base for showing related content of viewing item.

Supporting multi-source

To support multimedia content type, we not only need multiple views for different media type but also need get content from multi-source. For example, we need Video on Demand (VOD) through stream media server, and we

* Supported by the National Hi-Tech Research and Development Program of China (863 Project). Contract Number: 2003AA118101.

need have more control over downloading through FTP. Also we need email for notification.

Supporting fine-grained access control

Web content has special actions such as downloading and publishing. As the amount of content increases, user should focus on most valued, recommended content. What can be published and recommended must be carefully determined. So, we need fine-grained access control.

Lightweight and easy to use

Traditional CMS often contains complex functions and many definitions to learn. It's always very hard for administrators or advanced users involved in management of content, especially for those who just know how to browse web pages and click links/buttons.

In this paper, we introduce our Lightweight Web Content Management System (LWCMS). It is used as a lightweight framework for multimedia content management and meets all above requirements.

This paper is arranged as following: In section 2, we make an overview of content management. Then, we propose our architecture. Section 4 to section 8 describe the content partition definition, configuration of multi-source, meta-data structure, content life cycle management, and collaboration on content between users. After that, we evaluate our system. Finally, we make our conclusion and figure the future work.

2. Content management overview

CMS provides convenient methods for user to organize, share, and search content. It also provide platform for communications among users.

There have been lots of products for CMS solution. As Michael McIntosh pointed out:" while the concept of content management is not new, the implementation of a content management project is still a challenge, one for which there are more products than processes [5]", we can figure a process to cover most of those products (Figure 1).

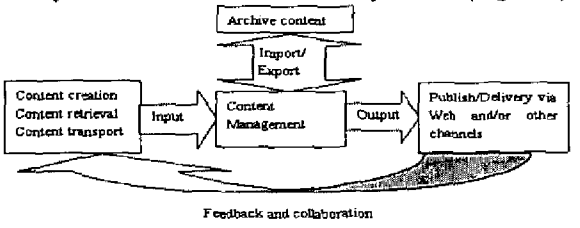

Figure 1: General process of a WCM

User can create content directly in CMS, or by specific media creation tool. Content also can be collected from other system, either by retrieval capacity of CMS, or by collaboration of CMS and outside application to transport content between them.

After content is inputted in the CMS, it will be managed by content administrators. They review and approve content suitable for publishing or deleting unvalued content. For large volume content, it also need import or export historic content by archive actions.

The approved content will be published and delivered on website for PC or mobile equipment via internet or wireless connection, they are available for browsing, viewing and downloading by users owned specific rights. Users make feedback and collaboration on website and it will be a source for input content.

3. Architecture

As the introduction section pointed out, we need to support several special requirements. Figure 2 gives a detailed view on LWCMS architecture. We layered the underlying system to end-user system from bottom to top. The blocks in gray are the outside systems we rely on.

At the bottom is the data source, data can be saved into or retrieve from database or file system. And for user related information we can adopt LDAP directory server for Single Sign On solution.

It is noticeable that web server, ftp server and stream media server are read-only to file system. We just accept writing by our own system and don't rely on other system's input method for our operation consistent and security purpose, because every outside system has its own style of operation and its own access control system. And there are well known leaks spread out on many servers. The multi-source support will be discussed in section 5.

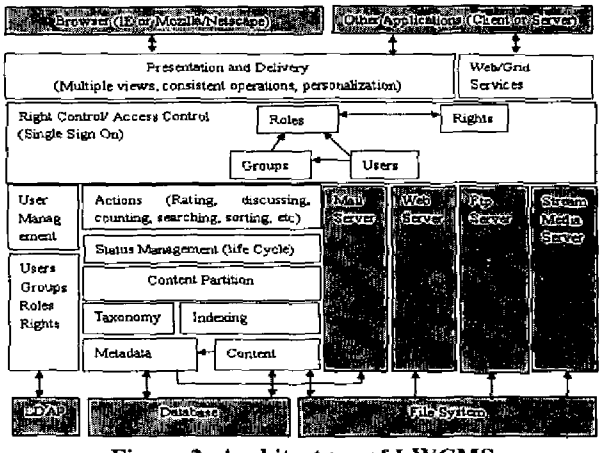

Figure 2: Architecture of LWCMS

In file system, the raw contents are saved in files for delivery by servers mentioned above. Although there are some multimedia databases, they process blob data in their own ways. Considered not relying on specific database system, our database only save text content and all content's descriptors (the meta-data).

Meta-data can be gained not only from user inputting, but also from automatically extraction by Information

Retrieval technology [6] (such as Feature Selection, Auto-digest) and Data Mining (for generate cross reference and other relationship between contents, such as Collaborative Filter). The meta-data schema conforms to a specific or self defined standard. We define an extendable data structure for this purpose. It will be described in section 6. For different content it can have the same schema uniform or various schemas for different view. Content meta-data can be subscribed and sent via email server.

To enhance the semantic for content, the meta-data generation can be combined with taxonomy to generate topic related descriptions and find relations between contents. The data and meta-data will be indexed for search and presented by sort and filter criteria. Of course, the indexed data also favors finding relations among contents using classify method or collaborative rules.

We introduce a new layer to separate contents, we name it content partition. It will be described in detail in section 4.

Till now, we reached the core part of a traditional CMS. A general CMS must provide functions for management of content. We call the life cycle of content as status switching. And in our system, we managed content not only itself but also its status.

Beyond the management of content and its status, our system provides functions for actions of end-user or other application such as rating the content, discussing about the content, online chatting about a specific topic, counting click number, scoring users, caching outside content and so on. And for all actions, we should control them by authentication and authorization. We'll describe the status management and access control on actions in section 7.

Since the presentation is for normal user search, browse and view content, it should be user friendly. Users need personalize on layout, color scheme or content. They also need transfer data from server to local machine through delivery channel of CMS.

As figure 2 shows, the components are clearly divided. Every module can be exposed as a service to construct service oriented architecture. Also, we can make the overall system pluggable by making clear slice of it.

4. Content partition

We need manage a variety of content. Though we can divide them by media type as MIME [7] does, it is not enough to reflect their usage. So we introduced a second layer, called "content partition". Every content partition may contain one or more "content type" as defined on MIME. Furthermore two special partitions are introduced. One is to stand for "Mixture or Any of above Types", called Courseware. To represent reference of content, we added link as another special content partition. All content partitions' definition is on table 1.

Table 1: Definition of Content Partitions

Media Type	Content partition	Description
Text	Bulletin/ Notice	We need bulletin/notice to notify user newly coming event.
	Message	For anonymous user to add comments for the entire website
	Notes	For registered user to add comments in forum or discussing a specific content on the site
	Article	Long text to delivery information, it can be viewed or edited online.
	Document	A piece of work created with a word processor. It often needs specific application to read and edit and appear as a file.
Audio	Audio	It can be music or voice in the site content scope. Users have rights can update their meta-data and/or play them online or download them.
Video	Movie	It can be movie or animation in the site content scope. Users' action on it is similar to audio.
Graphic	Graphic	Any graphic can be viewed on the web. Users having rights can update them and/or see them online.
Application	Software	Any binary file stands for tools needed to view content or other purpose.
Reference	Link	For reference of content on the web, exists as URI.
Mixture Types	Courseware	Any other information can't be contained by above or anything for purpose of education. It has no restriction on media type

For almost every content partition, we can associate several content types.

5. Supporting multi-source

To support multi-source content upload, we used an XML file for configuration. Several options could be used to configure for every content partition related to file system. They are defined as sub nodes of "upload" node.

Almost every node has sub nodes as Table 2 shows.

Table 2: Node interpretation

Node name	Description
pathString	Specify path to save file. It can be relative or absolute.
relative	If the sub node "relative" is true, then, the path is relative to "root" node.
maxSize	Specify the Maximum bytes to upload; the unit is 1K bytes.

view	Specify where to view the content. Often, it is a URI to specify content that can be view directly on web.
download	Specify where to download the content. Often, it is a URI to specify content for downloading.
extension	Specify a file extension list to constraint on which kind of content can be uploaded in the content partition.

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
- <config>
  - <upload>
    + <graphic>
    - <audio>
        <pathString>audio/</pathString>
        <view>rtsp://192.168.0.111:554/rescenter/audio/</view>
        <download>ftp://192.168.0.111/rescenter/audio/</download>
        <extension>rd,rm,au,midi,au,midi,mid,mp2,mpg,wav,wmv,wma,mpeg,rm,avi</extension>
        <!-- by default is relative to upload/root -->
        <relative>true</relative>
        <maxSize>100000</maxSize>
      </audio>
    + <software>
    + <video>
    + <documents>
    + <courseware>
    + <favorite>
    - <root>
        <pathString>/usr/local/realserver/Content/rescenter/</pathString>
        <!-- specify if it is relative to website root -->
        <relative>false</relative>
        <maxSize>100000</maxSize>
      </root>
    </upload>
  - <mail>
      <smtpserver>sel.buaa.edu.cn</smtpserver>
      <admin>Webmaster@sel.buaa.edu.cn</admin>
    - <subscribe>
      <authentication>true</authentication>
      <username>webmaster</username>
      <password>*****</password>
    </mail>
  + <information>
</config>
```

Figure 3: Content upload and delivery option configuration file of LWCMS

We can see from figure 3 that the "audio" node's path string is relative to the "root" node's path string. That is, the absolute file path for storing audio files is "/usr/local/realserver/Content/rescenter/audio/" according to interpretations on table 2. By using this XML file, we can configure multi-target saving and adding constraints on uploading (For audio, the max size allowed for uploading is 100M and the extension is constrained). We also can configure multi-source viewing or downloading (from stream media server or ftp server).

6. Use of meta-data

There are many meta-data standards for description and exchange purpose. Each standard are focus on a specific topic area. IMS [8] is for learning materials and MPEG-7 is mainly for video and audio [9]. Dublin core (DC) [2] is the well-known standard for general-purpose description of web content. It is suitable to describe any content in spite of its media type. But we usually need to focus on specific meta-data for different media type; DC is too common to achieve this. Because DC has 15-core elements and can't extend for more description; we need extra field for different media type. Just like RDF [10] does, we take the mechanism to contain DC as our core meta-data, and implement ours extendable by different media type.

To avoid confusing with Dublin Core's content element, we name a general content as resource (Figure 4). For simplicity, we call content partition as channel. Also we use category instead of taxonomy to map Dublin Core's corresponding element. It is the classification for any channel.

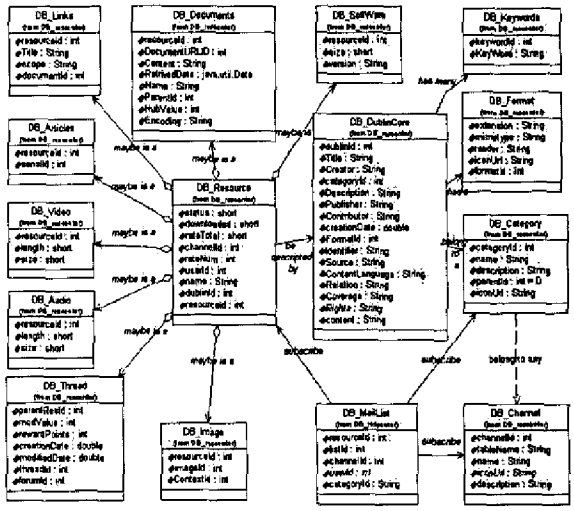

Figure 4: Entity relationships in LWCMS

We can see from figure 4 how we organize our data. Every content partition (including link, article, video, audio, image, document, thread and software) represent content which has its common features (such as click counting, download counting, status etc.) in resource object. A resource object also contains a Dublin core reference. Dublin core's keyword and category are maintainable. Category represent taxonomy, they are in hierarchy. For every channel, there is the same taxonomy.

By using "Maillist" object, user can subscribe content in specific channel, or in specific category of specific channel or just comments of specific content.

7. Content life cycle and access control

In our application, content is created either by specific application or directly on web, then it starts its life cycle from "start point" as Figure 5 shows.

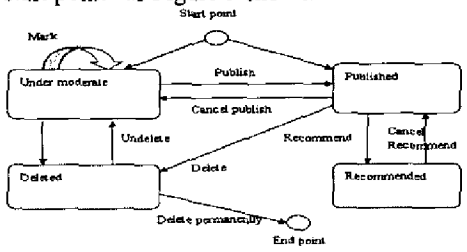

Figure 5: Life cycle of Content in LWCMS

The entire life cycle consist of four major states: Under moderate, Published, Recommended and Deleted.

Only published content can be seen by normal users. Deleted content can be recovered by administrator. Recommended content will be organized in special view for users' focus. Content in "Under moderate" state can be marked for future determination.

We control all rights in the process of status change. We assigned user and user groups for one or more roles, while roles can be assigned one or more rights according to content partition and taxonomy. We defined 5 kinds of rights to assign for roles. Those rights involve adding, editing, deleting, downloading, and publishing (Figure 6).

Roles with add and edit right can append and modify content in the channel or specific category. If the user hasn't "publish rights", the added or edited content will be in under-moderate state. Roles with downloading/deleting right can download/delete content. The deleted content can be recovered by administrator, only administrators can purge content. Roles with publishing right can publish content by adding or editing content. They also can recommend content in the specific scope.

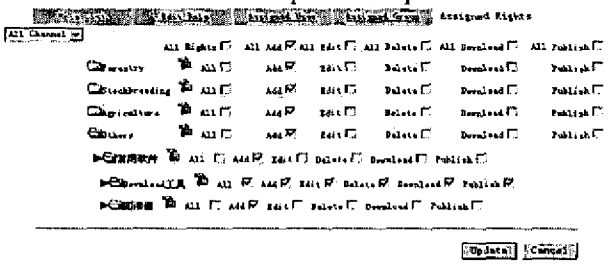

Figure 6: rights assignment in LWCMS (partial)

8. Interactions in LWCMS

To achieve usability goal, we developed a set of custom tags as web controls to keep UI consistence. And we use struts [11]'s validation framework to validate input.

They are many methods to interact among users. We support nearly every kind of them based on access control.

In addition to browsing, searching, viewing, playing, subscripting, and downloading from the website, registered users can help administrators do the content management, such as making comments for content, rating content, discussing on forum, and chatting online.

Because downloading and playing multimedia content consume much resource of server, so we put access control on these actions as we have seen in the previous section. Others actions can be done by or registered user by default.

For every registered user, they can do personalization for color scheme, layout for the whole site. They also can customize the content of the user specific home page.

Administrator can do anything normal user can do. Also they can coordinate chat rooms, manage rights and content in batch mode.

Anonymous user can browse and view some content. They also can leave messages for the website on message board.

9. Evaluation

There is an example to do evaluation on CMS by filling a large feature table [12], but it just suitable to prove that it conforms to most features needed. We can prove that also. Because every system has unique features and there are too many differences among them on usage even on same type of function, it's not so easy to compare them.

We mainly concern on providing a lightweight novel system applied our original ideas (conformed to the requirements mentioned in section 1). In compare to commercial enterprise CMS (such as Documentum, Vignette etc.), we lay less emphasis on performance optimization because we believe future adoption of grid technology will mend these. And because we target this system on non-technique users, we don't provide additional APIs for secondary development and provide relative less configuration options.

By combining all designs presented in above sections, we highlight benefits for normal user, content administrators, system administrators and programmers to evaluate our ideas.

Benefit for normal user

By using content partition, users can focus on specific media type based on usage. They can browse them by taxonomy in a specific channel and can find related content on other channels (as Figure 7's left panels show). They can focus on specific taxonomy to access all media types of the system. They also can get different views and descriptions of various media type in favor of extendable meta-data, and they can search content by Dublin Core element on all media type on integrated search.

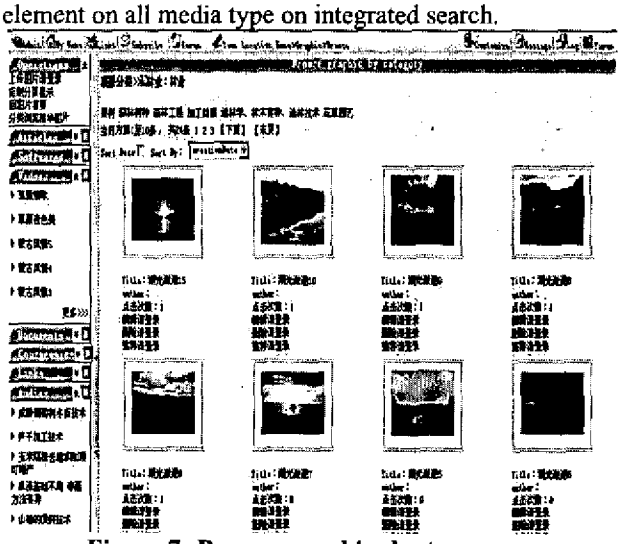

Figure 7: Browse graphics by taxonomy

They can fire actions on content by any type of communication method (see section 8). Operations they can do will automatic displayed on web pages based on rights assignment. The rights are taxonomy based, so they can focus on management of their expert field.

Benefit for content administrators

They have more rights and can do management in batch mode in administrator's page. Additionally, as Figure 6 shows, they can assign rights by roles and edit user/group role mapping easily as they wish.

Benefit for system administrators

They can deploy the system on any type of operation system supporting java, any relational database, and use any type of ftp sever, stream media server to deliver multimedia content by file by one step configuration on the XML file. Further more, they can distribute content on several machines and setup media server separately. It's easy to do that than using cluster technology.

Benefit for programmers

Programmers we referred here are those who are in charge of enhancing this system, not the end-user. They can follow our principles to add new type of content partition easily, and they will gain integrated search and taxonomy management capacity automatically.

They can expose new web service or grid service based on our basic functions for outside application usage. Also they can plug new module to retrieve meta-data from content without interfering the system's functions.

10. Conclusion and future work

We proposed our LWCMS in this paper. It is designed as a loose-coupled extendable architecture and has been developed from scratch to supporting multimedia management, taxonomy and role based fine-grained access control. In LWCMS, we defined content partition and extendable meta-data schema for multimedia management and presentation. And a configurable way is provided for multi-source content input and delivery support. Furthermore, LWCMS has been applied for agriculture, stockbreeding and forestry technology multimedia content management [13]. There are about 1000 multimedia content in it now and it has passed stress test. We also have made fully test on its functionality and usability. Our test team produced 831 software problem reports including inconvenient usage. We solved all effective problems. As a result, the system becomes stable for reality use. We now have a stable base for future enhancement.

In order to provide single sign on (SSO) solution for sites collaboration, we can adapt grid technology. Grid Security Infrastructure (GSI) [14] from Globus community defined delegation mechanism of authentication information.

Further improvements include retrieve content relationship by automatically content classification. There are many existing work showed us good results on classification and retrieval of research paper [15]. In order to adopt those algorithms on common content, we need hard work on this topic.

References

[1] Perrey, R. and Lycett, M., "Service-oriented architecture" Proceedings of 2003 Symposium on Applications and the Internet Workshops, 2003, pp. 116-119.

[2] Dublin Core Meta-data Element Set http://www.dublincore.org/documents/dces/

[3] RSS 2.0 specification, http://blogs.law.harvard.edu/tech/rss

[4] Vasudeva Varma, "Building Large Scale Ontology Networks", Proceedings of Language Engineering Conference, 13-15 Dec. 2002, pp: 121 – 127.

[5] Michael McIntosh, "Content Management Using the Rational Unified Process", rational Inc white paper, 2002

[6] Christos Faloutsos, Douglas W. Oard, "A Survey of Information Retrieval and Filtering Methods", Computer Science Technical Report Series, University of Maryland at College Park College Park, MD, USA , 1995, pp. 23

[7] Network Working Group, "RFC 2046 (rfc2046) - Multipurpose Internet Mail Extensions (MIME) ", http://www.faqs.org/rfcs/rfc2046.html

[8] IMS standard http://www.imsproject.org/

[9] Jane Hunter, "Enhancing the Semantic Interoperability of Multimedia Through a Core Ontology", Circuits and Systems for Video Technology, IEEE Transactions, Volume: 13, Issue: 1, Jan. 2003, pp. 49 - 58

[10] RDF standard http://www.w3.org/RDF/

[11] Struts project http://jakarta.apache.org/struts/index.html

[12] Norman W.Y. Shao, Stephen J.H. Yang, Addison Y.S. Sue, "A Content Management System for Adaptive Learning Environment", Proceedings of Fifth International Symposium 2003 on Multimedia Software Engineering, 2003, pp. 206 - 214

[13] Agriculture, stockbreeding and forestry technology information platform http://www.nmlpt.com

[14] Von Welch, Frank Siebenlist, Ian Foster, etc. "Security for Grid Services", Proceedings of 12th IEEE International Symposium on High Performance Distributed Computing, 2003. 22-24 June 2003, pp. 48 - 57

[15] Steve Lawrence, Kurt Bollacker, C. Lee Giles, "Indexing and Retrieval of Scientific Literature", Proceedings of the eighth international conference on Information and knowledge management, ACM Press New York, NY, USA, 1999, pp: 139 - 146.