Moodle Integrated Command Structure

Don Gable, PhD
Computer Information Systems Department
University of Dubuque
Dubuque, IA 52001
DGable@dbq.edu

Abstract

This paper describes the design and implementation of an integrated command structure that provides a rich, convenient method to link to any number of web applications from a Course Management System (CMS) such as Moodle using a common interface. Depending on the classification of the Moodle user (student, faculty, administrator) different applications will be accessed when clicking on the same link. Students will invoke one application; faculty will invoke a second application; administrators invoke a third application.

An example of this is the University of Dubuque Class Evaluation application. Each Moodle page contains a link to this interface structure. When a user with student credentials accesses the link they are directed to an application that displays the evaluation form to be filled out and submitted. When a user with faculty status accesses the link they invoke an application that permits them to view the survey results of the current course in a variety of formats or to download the results into an Excel spreadsheet. When an administrator accesses the link, they are directed to a page where any class can be selected from a dropdown list and the results viewed or downloaded.

The four applications that have been implemented using this technology at the University of Dubuque are described.

1 Introduction

The University of Dubuque (UD) has identified a number of applications that need to be accessed by most classes across the University. Since nearly all classes at UD are required to use the same Course Management System (CMS), the logical approach was to develop an integrated command structure to access these applications that could be uniformly applied to these classes with minimum implementation and maintenance effort.

The current applications include an Attendance Registration system, a class evaluation system, an electronic rubric (eRubric) system, and an implementation of the IAS class evaluation form.

2 Interfacing with the Course Management System

The CMS used at UD is the Moodle Course Management System. This open source CMS has been in use for approximately 4 years. Whereas all classes are required to have a Moodle site, some use their site only for administrative purposes such as a repository for the class syllabus and a vehicle for class announcements. This integrated command structure now makes it possible to also provide additional functionality.

Moodle provides a number of resources that can be placed on a classes' page. The 'Link to a file or web site' resource is the one used by this system.

This resource, when linking to a web site, requires a Uniform Resource Locator (URL), and can be programmed to pass up to five parameters to the URL. These parameters must be selected from a Moodle-specified list of possible constants. These constants are derived from three data sources:

- The user's profile information
- The specific class information
- Miscellaneous information: Institution-specific information

The task in designing an integrated command structure is to exploit this parameter passing.

3 Designing the Interface Structure

The server side scripting must be capable of verifying the source of the access, the identity of the user, the milieu of the access—the class, and the purpose of the access—the application to be run.

To this end, two of the five parameters are used to validate that the request originated from a UD Moodle page. A third parameter identifies the user. A fourth identifies the class.

The Moodle scheme of parameter passing does not support user-defined *values*. However, all parameter *names* are user defined. Leveraging this characteristic, one of the parameter names is encoded to include a 'command' value that indicates the specific application to run on the server.

s = Encrypted Code u =User – Username c = Course ID Number m=User User id r[nn] = Server URL

Table 1: URL Request Parameters

The parameter names (see Table 1) are user defined and are related to the particular Moodle-defined parameter values.

The ‘s’ parameter passes in a Moodle designated encrypted code that is a MD5 hash value of the user’s IP address concatenated to a secret string. This is used to validate that the request originated from the actual Moodle site.

The [nn] reference in the ‘r’ parameter name encodes the command number, e.g., r27, that will trigger transfer to the desired application with command code of 27. See the routing table description below.

The URL encoded into the Moodle resource is the application server where a routing script interprets the parameters, validates them, and passes control on to the appropriate application.

4 Operational Functionality

All applications are implemented as PHP scripts running on a dedicated virtual server. A MySQL database, termed the Academic Central or ADC database, is used to provide information to validate the input and to implement the functionality of the system through a series of stored procedures.

4.1 Validating the Parameters

In order to control access to the applications , the system must verify the class controlling the access, the user requesting the access, and application necessary to satisfy the request.

4.1.1 Identifying the Class

Each class has a unique classid assigned that identifies the school term, the department of the course, the course, and the section. For example, 201130CIS10101 indicates that the term is 201130 (Spring 2012, the department is CIS, the course is 101 and the section is 01.

The Course ID Number passed in as the ‘c’ parameter from Moodle provides this classid.

It is also possible to encode the classid in a term neutral and even a course neutral manner as shown below in Table 3.

4.1.2 Identifying the User

Each user of the system, students, faculty, and administrators has an entry in the user table. With primary key of the users’ UD logon ID, all relevant information including a security token that encodes each user’s role: Student, faculty, or administrator.

4.1.3 Determining the Application to Run

Once the routing script has validated the accessing application and user, it must then decode the r[nn] reference in the context of the specific Course ID Number by accessing a routing table in the controlling database.

The [nn] value maps to the cmd field in the router table.

Field	Type
classID	varchar(32)
cmd	int(11)
routeToFac	varchar(255)
routeToStu	varchar(255)
routeArg	varchar(255)
routeToAdm	varchar(255)

Table 2: Router Database Schema

The routing script then validates the user by querying the user table in the database. It then decodes the security token for the user to determine what role they represent so the appropriate application target script can be identified in the router table and executed.

classid	cmd	routetofac	routetostu	routetoadm
201110PHL35501	1	AttendanceManMon.php	RegisterUserE.php	AttendanceManMon.php
201130CIS10105	3	AttendanceRosterB.php	AccessDenied.html	AttendanceRosterB.php
20111000000000	80	iasFaculty.php	iasStudent.php	iasFaculty.php
00000000000000	90	UDCourseEvalFaculty.php	UDCourseEvalStudent.php	UDCourseEvalAdmin.php

Table 3: Router Table (Sample)

5 Maintaining Application Data

Application data is maintained in tables within the ADC database. These tables are transient and are dropped when the application is no longer active. For example, student class attendance data is recorded in the attendance table which will be emptied after the end of the current school term.

For applications such as the course evaluation application, that are essentially data gathering apps, the software provides a facility for downloading the data into Excel spreadsheet format.

6 Applications Currently Implemented

6.1 UD Course Evaluations

The Course Evaluation application provides a questionnaire with 10 questions determined to be relevant by the office of Academic Affairs.

When students access the Moodle link, they are directed to the questionnaire itself. Faculty access is directed to a module that can view the questionnaire data in either a form-by-form view or a aggregated view that summarizes the results. The data can also be downloaded into an Excel spreadsheet. Administrators access all data from a dropdown list of all possible classes.

The application requires that the data be collected anonymously while the fact that a student has or has not filled out the questionnaire is available to the instructor. This requires that the data be kept in an evaldata table which contains no user identifying information as to its source. Access to this data is restricted from faculty until after course grades have been submitted. However, students maintain access to their own questionnaire data via a surrogate key contained in their enrollment record for the class. Once final grades have been submitted as indicated by the term_grade_date fields in the term table, the enrollment records are modified to set the surrogate key fields to null and thus permit faculty access to the unidentifiable questionnaire data.

6.2 Attendance Registration

The attendance registration application can operate in two different modes. For classes that operate in classrooms with computer technology, the students' link transfers to a self-registration module.

Since the registration module can automatically control access by inspection of IP addresses, students are restricted to self-registering only from their actual classroom. Students attempting to register attendance from other locations are automatically registered as 'unexcused absence' and a notifying email is sent to the instructor announcing this attempt. Also, since the enrollment is programmatically enabled from 6 minutes prior to class until 12 minutes after class begins, students can be marked tardy.

Since IP addresses are being tracked, the attendance reports display the actual seat in the classroom from which the student registered.

If the class is not held in a computer lab, the link transfers to an Attendance Report showing the student their term-to-date attendance record. Faculty are transferred to a page where they can manually record attendance/absence/tardy information.

6.3 eRubric System

The eRubric System provides a page where students can see the grading criteria for some particular assignment. The instructor is shown a dropdown list of all students in the class. The instructor selects a student and then can provide grades for the rubric categories.

Once the grade has been recorded by the instructor, the student then can see the completed rubric form.

6.4 CIS Majors Vita

This application is an example of a quick-to-implement questionnaire that is being used to gather information from UD CIS majors that will assist CIS department faculty in identifying students to be honored in the annual Spring Honors Convocation. The questionnaire went up quickly and will come down once the necessary information is gathered.

When clicking on the link, students see the questionnaire. Faculty, clicking on the link see the students' information on a form-by-form basis.

7 Future Work

A number of new applications are being considered.

7.1 Advisee Satisfaction Survey

The University of Dubuque currently has a paper based system that advisors use to poll their advisees to gauge satisfaction with the advising process. This would work in the same manner as the course evaluation system described earlier. The student accesses the survey form and can complete it. The advisor then can access the aggregated information or can view individual surveys. Anonymity is guaranteed by limiting advisor access until a certain programmable threshold number of surveys have been submitted.

7.2 Early Warning System

The University of Dubuque plans to implement an early warning system that can integrate information from the attendance registration module with information from a new application that enables faculty and staff to create 'incidents' or 'cases' based on system-generated data (poor attendance) and faculty/staff observations. Such cases would be tracked by the system and would provide automatic notifications to stakeholders about events or milestones of interest.

The Moodle Integrated Command Structure would permit categories of faculty, staff, and students to have appropriate access to case records.

8 Other Enhancements

8.1 Improved Administrative Controls

Modifications are underway to streamline the implementation interface for applications within Moodle class pages.

A generalized class identifier will soon take the place of specific class-by-class identifiers that have limited the scalability of broad based applications such as attendance reporting or class evaluation surveys.

For example, initially the Attendance Registration application required a specific row in the Router table for every class and section using that application. A typical classid was '201130CIS10103' where 2011 is the school year, 30 is the term (Spring) and CIS10103 indicates CIS 101 section 03, Intro to Computers. The evolving identifier approach is to create term-specific classids (20113000000000) matching any class in the Spring term, or anonymous classids (00000000000000) that will match any class in any term. See Table 3.

The Moodle Integrated Command Structure is an evolving endeavor.

Appendix
Data Dictionary
The Academic Central (ADC) database is comprised of 7 controlling tables

Tables_in_adc
course
enrollment
mtime
router
term
udclass
user

Table A1 Tables in the ADC Database

Other tables are present for application data but are not permanent.

Course Table
The course table contains the course numbers and descriptions of all courses offered.

Field	Type	Null	Key	Default	Extra
courseID	int(8)	NO	PRI	NULL	auto_increment
courseNo	varchar(12)	YES		NULL	
courseDesc	varchar(48)	YES		NULL	

Table A2: Course Table

courseID	courseNo	courseDesc
1	CIS 101	Intro to Computers
5	CIS 215	Programming I
10	CIS 332	Database Systems I
11	CIS 371	C++ Programming
17	PHL 111	Intro to Philosophy
19	PHL 214	Environmental Perspectives
20	PHL 216	Business Ethics

Table A3: Course Table Sample Data

Class Table
The Class table contains of all classes held at UD.

Field	Type	Null	Key	Default	Extra
classID	varchar(32)	NO	PRI	NULL	
courseID	int(11)	YES		NULL	
termID	int(11)	YES		NULL	
sectionID	int(11)	YES		NULL	
mtim	int(4)	YES		NULL	
openFlag	int(1)	YES		NULL	
instrID	varchar(12)	YES		NULL	
instrEmail	varchar(48)	YES		NULL	

Table A4: Class Table

classid	courseid	termid	sectionid	mtim	instrid
201030CIS10103	1	201030	3	3	DGable
201030CIS10105	1	201030	5	16	DGable
201030CIS30501	8	201030	1	5	DGable
201030CIS31501	9	201030	1	6	DGable
201030CIS49565	16	201030	65	0	DGable
201030HEA24660	12	201030	60	0	MBelmont
201035CIS10101	1	201035	1	4	DGable
201110CIS10104	1	201110	4	9	KKessler
201110CIS10105	1	201110	5	16	PHenkes
201110CIS21501	5	201110	1	6	DGable
201110CIS33201	10	201110	1	5	DGable
201110EDU43201	104	201110	1	24	BLee
201110EDU43203	104	201110	3	24	SDeWeerdt
201110EDU43204	104	201110	4	24	SDeWeerdt
201110EDU43205	104	201110	5	24	TRyan
201110EDU43206	104	201110	6	24	TRyan
201110EDU49001	103	201110	1	24	JEtienne
201110EDU49002	103	201110	2	24	ABrandel
201110EDU49501	102	201110	1	24	DStork

Table A5: Class Table Sample Data

EnrollmentTable
The Enrollment table maps the many-to-many relationship between students and classes.

Field	Type	Null	Key	Default	Extra
enrollid	int(8)	NO	PRI	NULL	auto_increment
classID	varchar(20)	YES		NULL	
udlogon	varchar(20)	YES		NULL	
iasSubmit	date	YES		NULL	
evid	int(11)	YES		NULL	
evalSubmit	date	YES		NULL	

Table A6: Enrollment Table

enrollid	classid	udlogon	evid	evalsubmit
17617	201125UDTH28101	VAllen	410	2012-01-24
17620	201125UDTH28101	KHall	376	2012-01-22
17621	201125UDTH28101	ALHolt	387	2012-01-23
17622	201125UDTH28101	alewis	364	2012-01-21
17624	201125UDTH28101	AWhite	NULL	NULL
17625	201125UDTH28101	bwiley	NULL	NULL
17626	201125UDTH28101	ChFreese	382	2012-01-22

Table A7: Enrollment Table Sample Data

MTime Table

The MTime table contains of all information about class meeting times.

Field	Type	Null	Key	Default	Extra
mtim	int(4)	NO	PRI	NULL	auto_increment
mtimDesc	varchar(36)	YES		NULL	
dkey	int(5)	YES		NULL	
starttime	decimal(4,2)	YES		NULL	

Table A8: MTime Table

mtim	mtimDesc	dkey	starttime
1	MWF 8:00-8:50AM	42	8.00
2	MWF 9:00-9:50AM	42	9.00
3	MWF 10:00-10:50AM	42	10.00
4	MWF 11:30-12:20PM	42	11.50
5	MWF 12:30-1:20PM	42	12.50
6	MWF 1:30-2:20PM	42	13.50
18	M 7:00PM-9:30PM	32	19.00
19	M 8:00AM-8:50AM	32	8.00
20	W 8:00AM-8:50AM	8	8.00
21	F 8:00AM-8:50AM	2	8.00
22	On-line Class	0	0.00
23	J-Term Class	0	8.00

Table A9: MTime Table Sample Data

User Table
The user table contains information identifying all users

Field	Type	Null	Key	Default	Extra
pid	int(8)	NO	PRI	NULL	auto_increment
udlogon	varchar(20)	NO		NULL	
mid	int(6)	YES		NULL	
udid	int(8)	YES		NULL	
advisorid	varchar(20)	YES		NULL	
cclass	varchar(20)	YES		NULL	
fname	varchar(24)	YES		NULL	
mname	varchar(24)	YES		NULL	
lname	varchar(24)	YES		NULL	
email	varchar(48)	YES		NULL	
pwd	char(40)	YES		NULL	
dla	date	YES		NULL	
sectoken	int(6)	YES		NULL	
dateexpire	date	YES		NULL	

Table 10: User Table

pid	udlogon	udid	fname	lname	sectoken
1	SBaule	67158	Sarah	Baule	161
5	CHillard	67546	Corey	Hillard	161
6	KHines	66200	Kirstyn	Hines	161
7	CHirt	70469	Conner	Hirt	161
9	TLamers	58822	Thomas	Lamers	161
10	ELindsey	66892	Eric	Lindsey	161
11	BLopez	65713	BobbieSue	Lopez	161
12	ZMay	71178	Zane	May	161

Table 11: User Table Sample Data

Term Table
The term table provides important dates for each term

Field	Type	Null	Key	Default	Extra
termID	int(8)	NO	PRI	NULL	
termStartDay	int(2)	YES		NULL	
termStartMon	int(2)	YES		NULL	
termStartYear	int(4)	YES		NULL	
termEndDay	int(2)	YES		NULL	
termEndMon	int(2)	YES		NULL	
termEndYear	int(4)	YES		NULL	
curTerm	int(1)	YES		NULL	
termGradeDay	int(2)	YES		NULL	
termGradeMon	int(2)	YES		NULL	
termGradeYear	int(2)	YES		NULL	

Table 12: User Table

ID	SD	SM	SR	ED	EM	EY	GD	GM	GY	curTerm
201010	30	8	2010	14	12	2010	NULL	NULL	NULL	NULL
201030	24	1	2011	11	5	2011	NULL	NULL	NULL	NULL
201110	29	8	2011	16	12	2011	NULL	NULL	NULL	NULL
201125	3	1	2012	20	1	2012	29	1	2012	NULL
201130	23	1	2012	4	5	2012	14	5	2012	1

Table 13: Term Table Sample Data