UMCM: One Institution’s Implementation Experience with Kuali Curriculum Management

Michelle Appel – Functional Lead | UMD, College Park
Joseph Drasin – Project Manager | UMD, College Park
Garey Taylor – Technical Lead | UMD, College Park
University of Maryland

- Large “flagship” University
  - 12 Colleges, over 100 departments
  - 150+ programs of study, depending on definition
  - 26,000+ undergraduates, 10,000+ grad students
  - ~8,000 employees including ~ 1,500 T/Tk faculty

- Growing Kuali Community on campus
  - Rice & COEUS in production
  - KFS & OLE in early stages

- Legacy, homegrown SIS on a mainframe
  - Challenges with ability to sustain expertise on technology
  - Difficulty continuing to meet changing campus needs
Project Introduction

- Team based in College Park, MD, South Africa, and Vancouver, BC
- KS 1.2 codebase
  - Fixed issues for 1.2.1
  - Going live with 1.2.1 with UM overlay
- Focused year 1 implementation on course proposal and approval
  - Create a course
  - Modify a course
  - Retire a course
  - Workflow
  - Authorization
  - Dependency analysis
Project Timeline

- Major Milestones
  - 7/21/2011 Project Kickoff
  - 8/26/2011 Milestone 0 “Infrastructure”
    - 3: two week sprints
  - 11/1/2011 Milestone 1 “The Plumbing”
    - 4: two week sprints (1 week QA)
  - 12/13/2011 Milestone 2 “Soft Launch”
    - 2: two week sprints (1 week QA)
  - 1/31/2012 Milestone 3 “Production Build”
    - 3: two week sprints (4 weeks UAT)
  - 3/7/2012 Go-Live

Where are we today
- Getting ready for our pilot colleges
- Localizing
- Retire Course by Proposal Feature
- Rules – Rules types, categories, entry
Organizational structure

- Large multi-disciplinary team
- Structural Evolution
- Current Structure
  - Data Team (Functional & Technical)
  - Infrastructure Team
  - Software Development Team
  - Functional Working Group (including UI)
  - Functional Rules Group
  - Quality Assurance
  - Documentation and Training
- Continual improvement
Development Methodologies

- Hybrid
  - Upfront Waterfall Style Requirements Gathering
    - Dedicated Business Analysts
    - Dedicated UX
      - Rapid Prototyping with Azure
  - Agile Style Development
    - Two Week Sprints
    - Daily Standups
    - Product Owner Priority Meetings
    - Sprint Retrospectives
Curriculum Management

Create...
Propose a new course or start a new program.
- Create a Course
- Create an Academic Program

Find...
Search for proposals, courses, and programs to view, modify, or retire.
- Courses
  - Browse Course Catalog
  - Find a Course
  - Find a Course Proposal
- Programs
  - Browse Academic Programs
  - Find Academic Programs
  - Find a Program Proposal
  - View Core Programs
  - View Credential Programs

Recently Viewed
View courses and programs that you have recently visited.
You have no recently viewed items.

Management Tools
Support tools for courses and programs.
- Course Set Management
- Learning Outcome Categories
- Dependency Analysis
KS Landing Page

New color scheme, navigation menu moved
CM Landing Page

Testudo
Curriculum Management

Home » Curriculum Management

Curriculum Management

Create...
Propose a new course or start a new program.

Create a Course

Find...
Search for proposals, courses and programs to view, modify or retire.

Courses
Browse Course Catalog
Find a Course
Find a Course Proposal

Programs
Browse Academic Programs
Find Academic Programs
Find a Program Proposal
View Core Programs
View Credential Programs

Recently Viewed
View courses and programs that you have recently visited.

You have no recently viewed items.

Management Tools
Support tools for courses and programs.

Course Set Management
Learning Outcome Categories
Dependency Analysis
Create a Course – Course Information

Transcript Title: Required on Submit
This is the truncated title that will appear on the printed transcript.
KD TEST
17 characters left

Description and Reason for Proposal

Description: Required on Submit
This description will appear in the catalog.

Additional Course Information
Provide any additional information about this course that does not fit in either the catalog description or course requisites.
Create a Course – Governance

KD Proposal (Proposal)
Proposal Status: Saved | Comments | Decisions

Governance

Curriculum Oversight Unit: Required on Submit
This is the unit(s) responsible for the content of the program. This selection will determine
the workflow/approval process for this proposal.

[Add to list]

SVPAAAP-Office of the Registrar
[Save and Continue] [Cancel]

Maryland data, constrained values
Create a Course – Course Logistics

<table>
<thead>
<tr>
<th>COURSE SECTIONS</th>
<th>Course Logistics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Course Information</td>
<td></td>
</tr>
<tr>
<td>Governance</td>
<td></td>
</tr>
<tr>
<td>Course Logistics</td>
<td></td>
</tr>
<tr>
<td>Learning Outcomes</td>
<td></td>
</tr>
<tr>
<td>Course Requisites</td>
<td></td>
</tr>
<tr>
<td>Active Dates</td>
<td></td>
</tr>
<tr>
<td>Collaborators</td>
<td></td>
</tr>
<tr>
<td>Supporting Documents</td>
<td></td>
</tr>
<tr>
<td>Review Proposal</td>
<td></td>
</tr>
</tbody>
</table>

**Course Logistics**

**Scheduling**

**Term**

Selecting a single term will restrict this course to only that term. *Any* will allow the course to be offered in any term that matches the duration selected below.

- [ ] Fall
- [ ] Spring
- [ ] Winter

**Grading Method**

**How will you grade this course?**

**Assessment Scale** *(Required on Submit)*

- [ ] Completed notation
- [x] Regular (Letter)
- [ ] Satisfactory/Fail

**Student Registration Options**

If a course has been approved with pass/fail and audit registration options, a department can determine on a term-by-term basis whether to schedule the course with those options. Therefore, it is recommended that departments include all registration options when seeking approval for a course.

**Audit**

- [ ] Allow eligible students to select audit option.

**Pass-Fail**

*Customized values*
Create a Course – Course Requisites

New Rule Types

Step 2: Combine Rules with Logic

Kuali
EMPOWERING THE COMMUNITY
Summary of Major areas of Work

- Data work – mapping, clean-up, loading
- Localization
  - UI
  - Syntax
  - Fields
- Authorization setup
- Workflow
- Rules
  - Configuration – types, categories
  - Data entry
- Retire a course
Lessons Learned - Management

- Management
  - Be Flexible
  - Prepare for the unexpected (technically and functionally)
  - High visibility, but low volume use and impact
  - Set manageable (small) scope
    - Set expectations
  - Stick to time box
  - System of record issues will need to be addressed
Lessons Learned - Functional

- Functional
  - Data work is a huge effort – start early
  - Sometimes you don't know enough to make the decisions
    - Development possibilities and directions
    - Changing campus needs and business processes
  - Software has features campus wants, but isn’t ready for
  - Timing of functional input can be challenging
    - “Hurry up and wait”
    - Dedicated analysis time
  - Find forgiving functional users – testers, pilot users
  - Be open to new collaboration tools, willing to adjust midstream
  - Difficult to balance timing of training and documentation development with software development
Lessons Learned - Technical

- Technical
  - Development Methodologies
  - Development Environment Setup
    - Development Tools (Confluence, Jira, Bamboo)
    - Local Overlay Project (Coding Environment)
    - Kuali Student 1.2.1
    - Server Environments
  - Data Load Testing
  - Authorization
Development Tools - Confluence

- Confluence – Wiki
  - Functional Documentation
  - Design Specs
  - Developer Documentation
Development Tools - JIRA

- **JIRA – Issue Tracking**
  - Functional Issue Tracking
  - Technical Issue Tracking

**Pie Chart: KSCM Ready For Development**

- Backlog = 65
- Sprint 2-1 = 40
- Unscheduled = 1

*Total Issues: 100  Statistical Type: Fix For Versions (all)*
Development Tools - Bamboo

- Bamboo – Continuous Integration
  - Test Code Compilation
  - Build Development Artifacts
  - Test Data Load: UM->KS
  - Test Deployment
  - Test UI via Selenium
Local Development Environment

- Local Overlay Project
  - KS 1.2.1 Base Dependency
  - 5 Local Sub Projects
    - Umd-cm-cfg-dbs – DBs
      - Baseline
      - Developer Reference
    - Umd-cm-impl – Impl Overrides
    - Umd-cm-rice – Rice Overlay
    - Umd-cm-ui – UI Overrides
    - Umd-cm-web – WAR Deploys
KS 1.2.1

- “Special” Collaborative Patch Branch
  - First Full Reference Implementation
    - Critical Issues Will Be Found
  - Collaboration Between UMD, NWU, and KS
Server Environments

- **Dev**
  - Nightly Build – Full Data Load

- **Public**
  - Nightly Build – Reference Impex Load

- **QA**
  - Manual Build – Full Data Load
  - Duplicate of Production Env
    - 4 KS App Servers Running Parallel

- **Production**
  - Manual Build
    - One Initial Load
    - Nightly Update Load
Data Load Testing

- Tool to Populate KS Implementation with UMD Data
- Loads into KS via Web Services
  - DB to DB was deemed to complex
  - Web Service Contract Stability
  - Low Volume
- Full Load run every night to test iterative changes
- Production loads will only update changes
Authorization

- KS 1.1 & 1.2 Partially Implemented Authorization
- Adding Standard UI Authorization framework
- Finer grained permission checking
Questions?

Michelle Appel – mappel@umd.edu

Joe Drasin – jdrasin@umd.edu

Garey Taylor – gpt@umd.edu