# OpenCms at The Royal Library

An implementation Story

## Presentation Overview

- Background
- Implementation process overview
- Why OpenCms/opensource
- Porting existing content/services
- Integration of Digital Asset Management System (Cumulus) and Image server (eRez).
- Giving back to the opensource community
- The KB Suite

## Organization

- National library
- Copenhagen University library
- Cultural department

# National Library

- Preserving and providing access to Cultural heritage
- Books, Journals, Manuscripts, Maps and pictures, Printed matter, Music and now the internet.
- Only available in library reading rooms – and often only to people with documented need to access
- Digitisation
- Wider availability

## Copenhagen University Library

- Library services to university students, -teachers and –researchers
- Books and journals
- Most material available for lending out of the library
- Providing access to licensed electronic material
- Providing library services online
- Courses on how to use library services
- Wider availability (electronic reference material)

## Cultural Department

- Cultural events
- Concerts
- Museums and exhibitions
- Talks
- Advertising events
- Electronic exhibitions

## The situation before CMS

- Random development with minimal coordination
- No established editorial procedures
- No common look and feel
- Huge technology stack
- Very varying quality
- A maintenance nightmare !

## Resources available

- Implementing CMS and building new web site
- 6 man years of internal labour for development
- 6-7 developers
- 3-4 editors
- 67000 euro budget (excluding internal labour costs)
- Time: Feb 1st 2006 to Jan 27th 2007

## The Process

- Feb 06 kickoff
- Apr 06 requirements specification
- Jun(early) 06 OpenCms chosen
- Jun(late) 06 first graphic design
- Jul 06 development starts
- Nov 06 user training
- Dec 06 CMS opens for content providers
- Jan(late) 07 CMS with new web in production
- Feb(late) 07 project closedown

# CMS selection

- What can a CMS do for us ?
- Market survey
- Requirements
- Selecting candidate CMS’s (CMD, Sitecore, Plone, OpenCms)
- Testing candidates
- Inviting companies/consultants to tell us how and how well their systems could fulfil or requirements.
- Intensive one day OpenCms workshop for entire department
- Recommending a choice to management
- Defending choice to political leadership

## Why OpenCms ?

- Budget allowed only low cost or free base CMS software
- Staff capable of in house development
- Fitted existing technology stack (java, Oracle, tomcat, apache)
- Needed high flexibility to handle new requirements as they emerge.
- Ease of integrating with other systems
- Potential to present as editorial system

## Customizing

- Close cooperation between developers and editors (users)
- Component concept
- Resource types – structured content
- Handling existing services
- DAMS for images
- Custom property dialog

## Porting existing services

- The good, the bad and the ugly
- All Oracle web applications
- Good – functional applications with own identity
- Bad – more or less functional applications running under the KB site
- Ugly – non functional applications
- Good continued unchanged
- Bad and Ugly – content (database) reused in new common application
- A lot of old content also continued unchanged outside the cms

## Images

- Digital Asset Management System (Cumulus by Canto)
- Image server (eRez by Yawah)
- Acquired independently from the CMS project (digital preservation project)
- New image dialog for WYSIWYG editor
- New widget for xml editor

## Giving back to the community

- Customisations made by the Royal Library all available as opensource (LGPL)
- Modules of general interest available for download now – source code and other modules available on request.
- Why giving back ?
- Self interest
  - Use by community lessens risk of being forced to change to other system.
  - Ambition to get other institutions under Danish ministry of culture to use our system
  - Potential integration in OpenCms core lessens risk of system being broken by future updates

## The KB suite

- Component concept
- Any resource can be a component
- Rendering method
- Different rendering methods in different contexts
- Different rendering methods for different resource types

## The KB suite

- Component types of the KB web (not actually part of the suite)
  - Content pages
    - Standard
    - Arrangement
    - News
    - Course
  - Front Pages
  - ”Pure” components
    - Spot
    - Fact Box.

## The KB suite

- Component types included in the suite
  - Default
  - Jsp/plain text
  - Collector configuration
  - (link) list
  - Agent
  - Email form

## The KB suite

- Navigation mechanisms
  - Menus
  - Inherited content

## The Results

- Happy users
- Consistent look and feel
- Component concept has proven itself
- Close developer-user relationship
- System maintenance could be easier (but much better than without CMS)
- Successful integration with other systems
- KB Suite released to the world
- Happy users !